

Changes in SPDX 3.0

Bob Martin
Senior Software and Supply Chain Principal Engineer
Cyber Solutions Innovation Center
MITRE Labs

23 January 2024



MITRE | SOLVING PROBLEMS
FOR A SAFER WORLD™

Market Transparency through “Software Bill of Materials”

- **Third party components are a known systemic risk.**
 - Transparency can drive tools and behavior to document risk, support mitigations, and drive better SW development practices.
- **NTIA at Commerce launched an open, community-driven, cross-sector “multistakeholder process” to promote software component transparency in 2018.**
 - Understand the problem and define basics of SBOM
 - Develop use cases across sectors on how such data can be used, today and in the future.
 - Guidance on how to use existing standards to implement SBOM
 - Software ID tags (SWID)
 - Software Package Data Exchange (SPDX)
 - CycloneDX
- **Deliverables <https://ntia.gov/sbom>**
- **Moved effort to DHS CISA (2021)...**
- **Deliverables <https://cisa.gov/sbom>**



CYBERSECURITY
& INFRASTRUCTURE
SECURITY AGENCY



NTIA Transparency Phase 1 Final Products

Framing Software Component Transparency: Establishing a Common Software Bill of Material (SBOM)

NTIA Multistakeholder Process on Software Component Transparency
Framing Working Group
2019-11-12



Eamonn Ó Muiri
<https://i1ic.kr/p/46dsiz>
<https://creativecommons.org/licenses/by/2.0/legalcode>

Roles and Benefits for SBOM Across the Supply Chain NTIA Multistakeholder Process on Software Component Transparency Use Cases and State of Practice Working Group

Introduction

The Software Supply Chain
About this document: Goals and Methodology

Perspective: Produce Software

- Reduce unplanned, unscheduled work
- Reduce code bloat
- Adequately understand dependencies within broader complex projects
- Know and comply with the license obligations
- Monitor components for vulnerabilities
- End-of-life (EOL)
- Make code easier to review
- A blacklist of banned components
- Provide an SBOM to a customer

Perspective: Choose Software

- Identify potentially vulnerable components
- A more targeted security analysis
- Verify the sourcing
- Compliance with policies
- Aware of end-of-life components
- Verify some claims
- Understand the software's integration
- Pre-purchase and pre-installation planning
- Market signal

Perspective: Operate Software

- Organization can quickly evaluate whether it is using the component
- Drive independent mitigations
- Make more informed risk-based decisions
- Alerts about potential end-of-life
- Better support compliance and reporting requirements
- Reduce costs through a more streamlined and efficient administration

Ecosystem, Network Effects, and Public Health Benefits of SBOM

- Accelerated Vulnerability Management

Survey of Existing SBOM Formats and Standards - Version 20191025

Survey of Existing SBOM Formats and Standards



Credit

NTIA Multistakeholder Process on Software Component
Standards and Formats Working Group
Final Version - 20191025

1

13

13

14

15

SOFTWARE COMPONENT TRANSPARENCY: HEALTHCARE PROOF OF CONCEPT REPORT

*Drafted as part of a process convened by the National
Telecommunications and Information Administration*

October 1, 2019

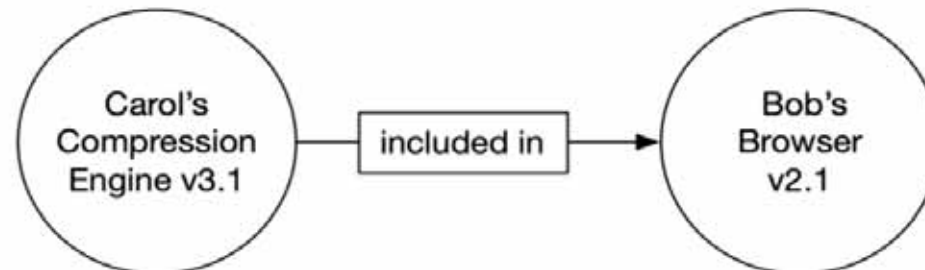
SBOM Definition

NTIA Minimal Elements (EO 14028)

Data Field	Description
Supplier Name	The name of an entity that creates, defines, and identifies components.
Component Name	Designation assigned to a unit of software defined by the original supplier.
Version of the Component	Identifier used by the supplier to specify a change in software from a previously identified version.
Other Unique Identifiers	Other identifiers that are used to identify a component, or serve as a look-up key for relevant databases.
Dependency Relationship	Characterizing the relationship that an upstream component X is included in software Y.
Author of SBOM Data	The name of the entity that creates the SBOM data for this component.
Timestamp	Record of the date and time of the SBOM data assembly.

https://www.ntia.doc.gov/files/ntia/publications/sbom_minimum_elements_report.pdf

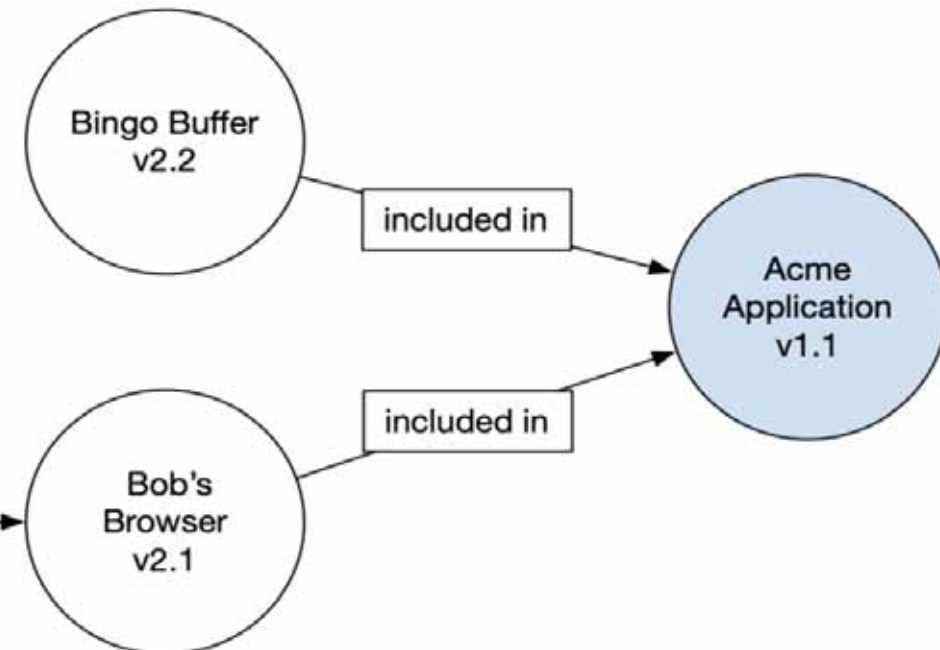
SPDX
CycloneDX
SWID



Source: https://www.ntia.gov/files/ntia/publications/ntia_sbom_framing_2nd_edition_20211021.pdf

Minimum Elements	
Data Fields	Document baseline information about each component that should be tracked: Supplier, Component Name, Version of the Component, Other Unique Identifiers, Dependency Relationship, Author of SBOM Data, and Timestamp.
Automation Support	Support automation, including via automatic generation and machine-readability to allow for scaling across the software ecosystem. Data formats used to generate and consume SBOMs include SPDX, CycloneDX, and SWID tags.
Practices and Processes	Define the operations of SBOM requests, generation and use including: Frequency, Depth, Known Unknowns, Distribution and Delivery, Access Control, and Accommodation of Mistakes.

https://www.ntia.doc.gov/files/ntia/publications/sbom_minimum_elements_report.pdf



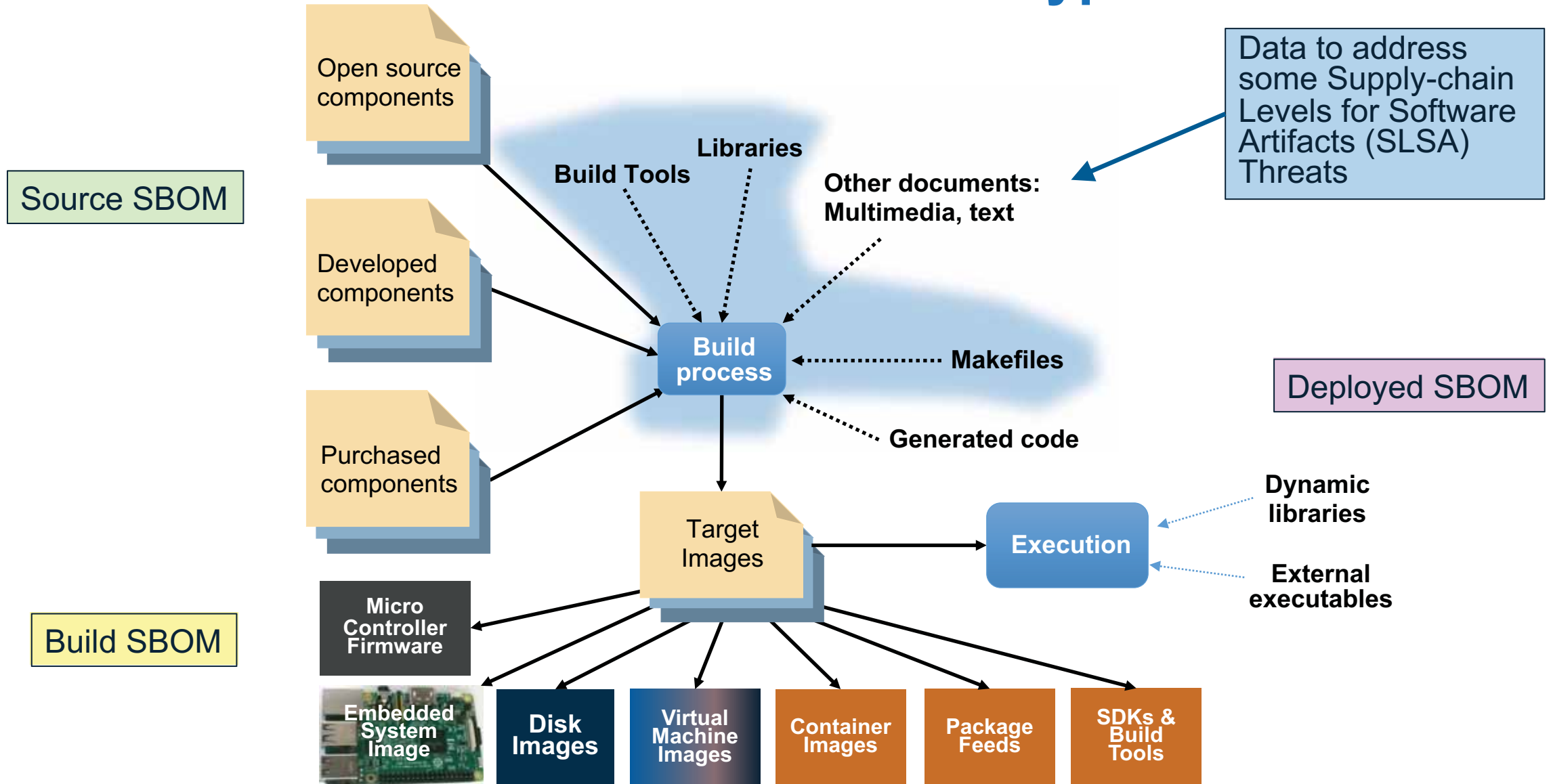
From the Community-led Working Group on SBOM Tooling and Implementation, facilitated by Cybersecurity and Infrastructure Security Agency [cisa.gov/sbom]



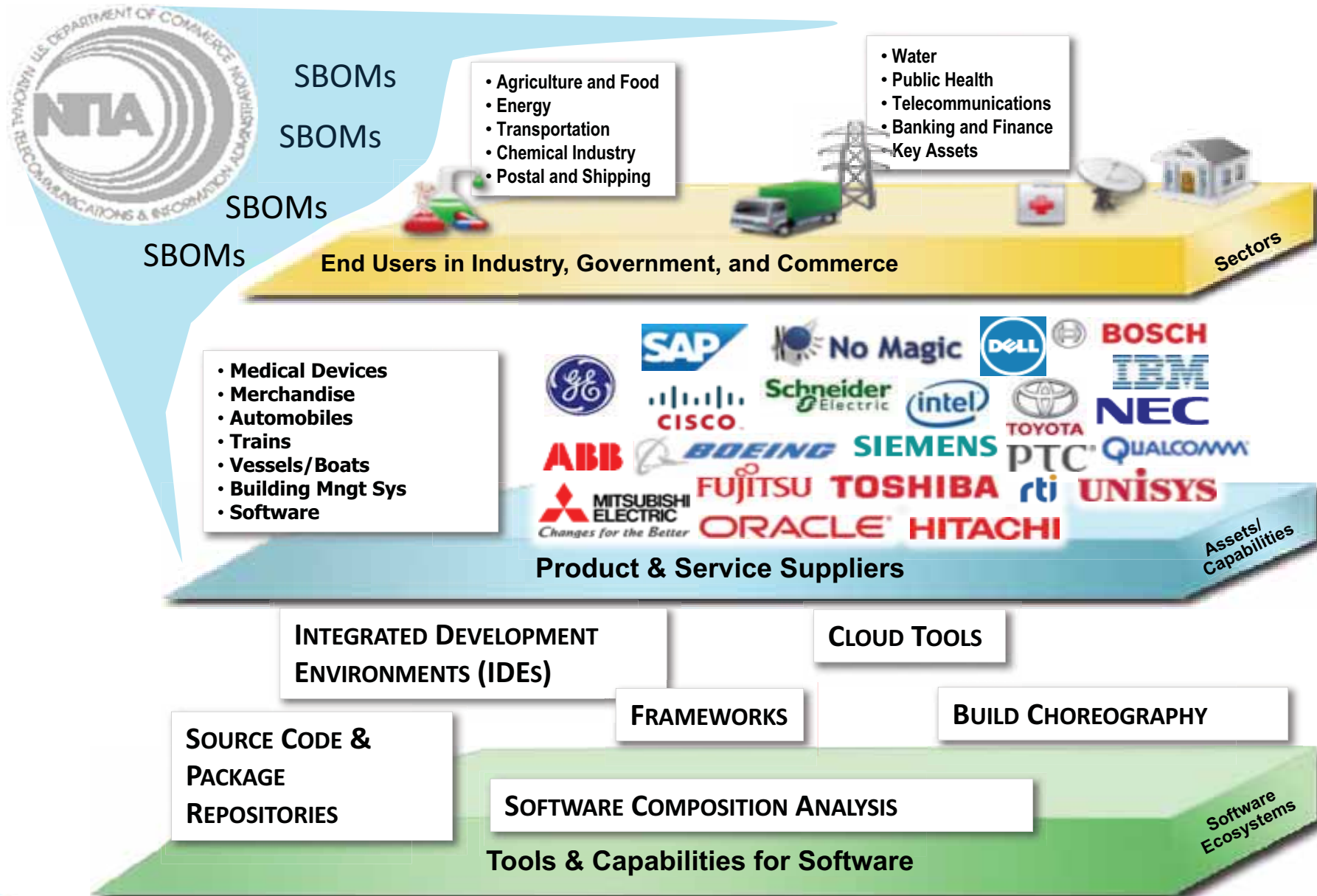
Table 1: SBOM Type Definition and Composition

SBOM Type	Definition	Data Description
Design	SBOM of intended design of included components (some of which may not exist) for a new software artifact.	Typically derived from a design specification, RFP, or initial concept.
Source	SBOM created directly from the development environment, source files, and included dependencies used to build an product artifact.	Typically generated from software composition analysis (SCA) tooling, with manual clarifications.
Build	SBOM generated as part of the process of building the software to create a releasable artifact (e.g., executable or package) from data such as source files, dependencies, built components, build process ephemeral data, and other SBOMs.	Typically generated as part of a build process. May consist of integrated intermediate Build and Source SBOMs for a final release artifact SBOM.
Analyzed	SBOM generated through analysis of artifacts (e.g., executables, packages, containers, and virtual machine images) after its build. Such analysis generally requires a variety of heuristics. In some contexts, this may also be referred to as a "3rd party" SBOM.	Typically generated through analysis of artifacts by 3rd party tooling.
Deployed	SBOM provides an inventory of software that is present on a system. This may be an assembly of other SBOMs that combines analysis of configuration options, and examination of execution behavior in a (potentially simulated) deployment environment.	Typically generated by recording the SBOMs and configuration information of artifacts that have been installed on systems.
Runtime	SBOM generated through instrumenting the system running the software, to capture only what is loaded and executing in memory, as well as external call-outs or dynamically loaded components. In some contexts, this may also be referred to as an "Instrumented" or "Dynamic" SBOM.	Typically generated from tooling interacting with a system to record the artifacts present in a running environment and/or that have been executed.

Software Bill of Materials Types



Enabling Adoption of SBOMs





CISQ/OMG Tool-to-Tool Software Bill of Materials Effort

Launched 24 Sep 2019

CHAIRS

- ▶ Bob Martin (MITRE)
- ▶ Dr. Bill Curtis (CISQ)
- ▶ Kay Williams (Microsoft - Azure - CD Foundation)

PARTICIPANTS

- ▶ Ken Modeste (UL)
- ▶ Philippe-Emmanuel Douziech (CAST)
- ▶ Santiago Torres-Arias (in-toto/NYU)
- ▶ Diahann Gooden (BlackBerry)
- ▶ William Cox (Black Duck by Synopsys)
- ▶ Steve Lasker (Microsoft - Artifact Storage, Open Container Initiative)
- ▶ Brian Russell (Google - CD Foundation)
- ▶ Nitesh Bakiwal (Microsoft - Windows)
- ▶ Kate Stewart (Linux Foundation - SPDX)
- ▶ William Bartholomew (GitHub)
- ▶ David Edelson (IBM - GCC)
- ▶ Jason Shaver (Microsoft - Developer Division)
- ▶ Thomas Steenbergen (HERE - OSS Review Toolkit)
- ▶ Sean Barnum (MITRE)
- ▶ Charles Schmidt (MITRE)
- ▶ Kriti Pandit (Microsoft - Windows)
- ▶ Michael Richardson (Sandelman Software Works Inc.)
- ▶ Sridhar Poduri (Microsoft - Supply Chains)
- ▶ Alexander Stein (Flexion)
- ▶ Deep Datta (JFrog)
- ▶ Shelly Waite-Bey (Waite SLTS - Cybersecurity, Compliance and AI)
- ▶ Dan Beard (Medcrypt - Heimdall)
- ▶ Manish Jadhav (Vigilant Ops - InSight)
- ▶ Gerald Heidenreich (Microsoft - CloudBuild)
- ▶ Vijay Chari (Radiometer IDC - Medical Devices)
- ▶ Allan Friedman (NTIA)
- ▶ Gerald Heidenreich (Microsoft - CloudBuild)
- ▶ Dan Lorenc (Google - CD Foundation)
- ▶ Jeffrey Martin (White Source Software)
- ▶ Michael Muller (CAST)
- ▶ Bryan Sullivan (Microsoft - Security and Compliance)
- ▶ Brian Fox (Sonatype)
- ▶ Fred Blaise (CloudBees - Jenkins)
- ▶ Mark Galpin (JFrog)
- ▶ Gary O'Neal (Source Auditor - SPDX)
- ▶ Anna Debenham (Snyk)
- ▶ Duncan Sparrell (sFractal Consulting)
- ▶ Philippe Ombredanne (nexB - ScanCode)
- ▶ Solomon Rubin (FOSSA)
- ▶ Adrian Diglio (Microsoft - Engineering Systems)
- ▶ Steve Winslow (Linux Foundation)
- ▶ Henk Birkholz (Fraunhofer SIT - IETF RATS and RIM)
- ▶ Paul Anderson (GrammaTech)
- ▶ Gareth Rushgrove (Snyk)
- ▶ Dick Brooks (Reliable Energy Analytics)
- ▶ Adam Boulton (Blackberry - Jarvis)
- ▶ Justin Cormack (Docker)
- ▶ Jack Conroy (Verizon)

<https://www.it-cisq.org/software-bill-of-materials/>

3T-SBOM Usage Scenarios Around SBOMs



Refer, Transfer or Purchase
(definition of what it is)

Pedigree
(history of how it was produced)

Provenance
(chain of custody of it)

Integrity
(cryptographic basis of unalteredness)

Proper and Legal
(conditions about its use)

Known Sw Vulns
(known fixes are applied to it)

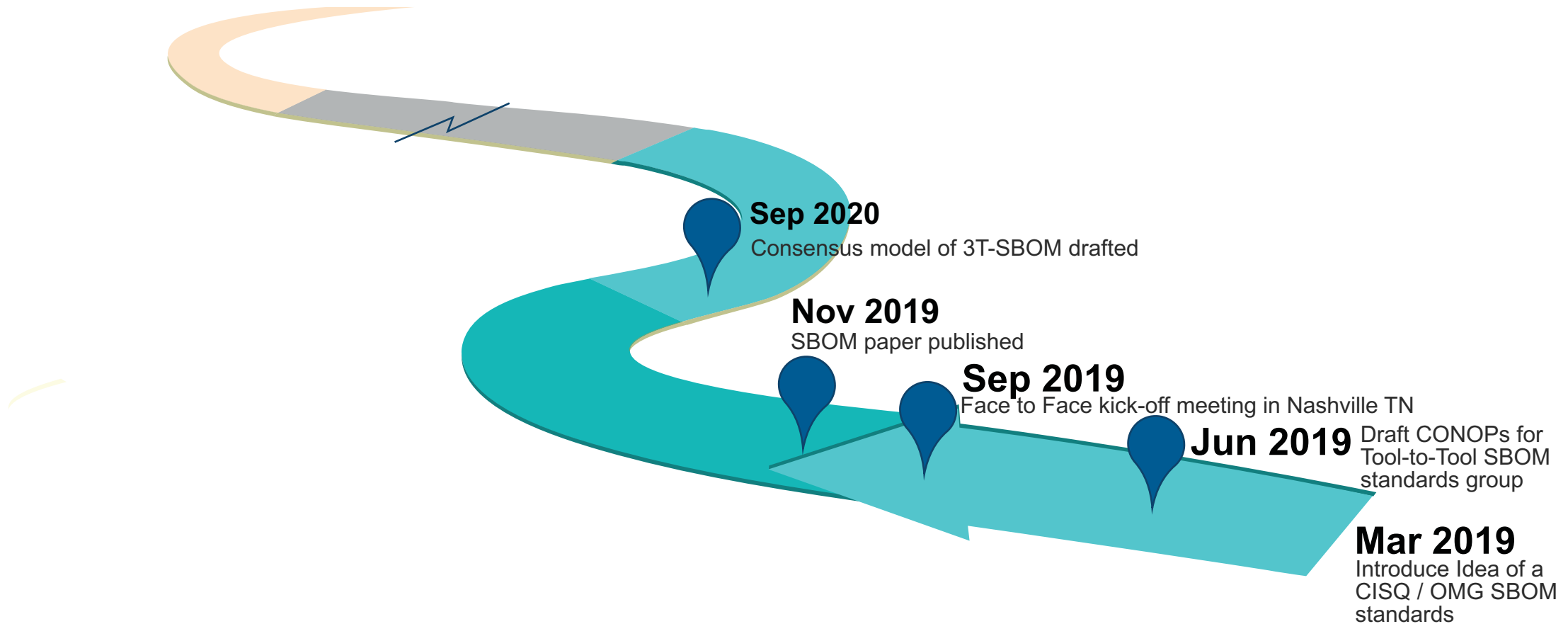
Assurance
(safe-secure-resilient)

SBoM of a SW Service
(SBoM of sw delivering service)

Supply Chain Sequence Integrity

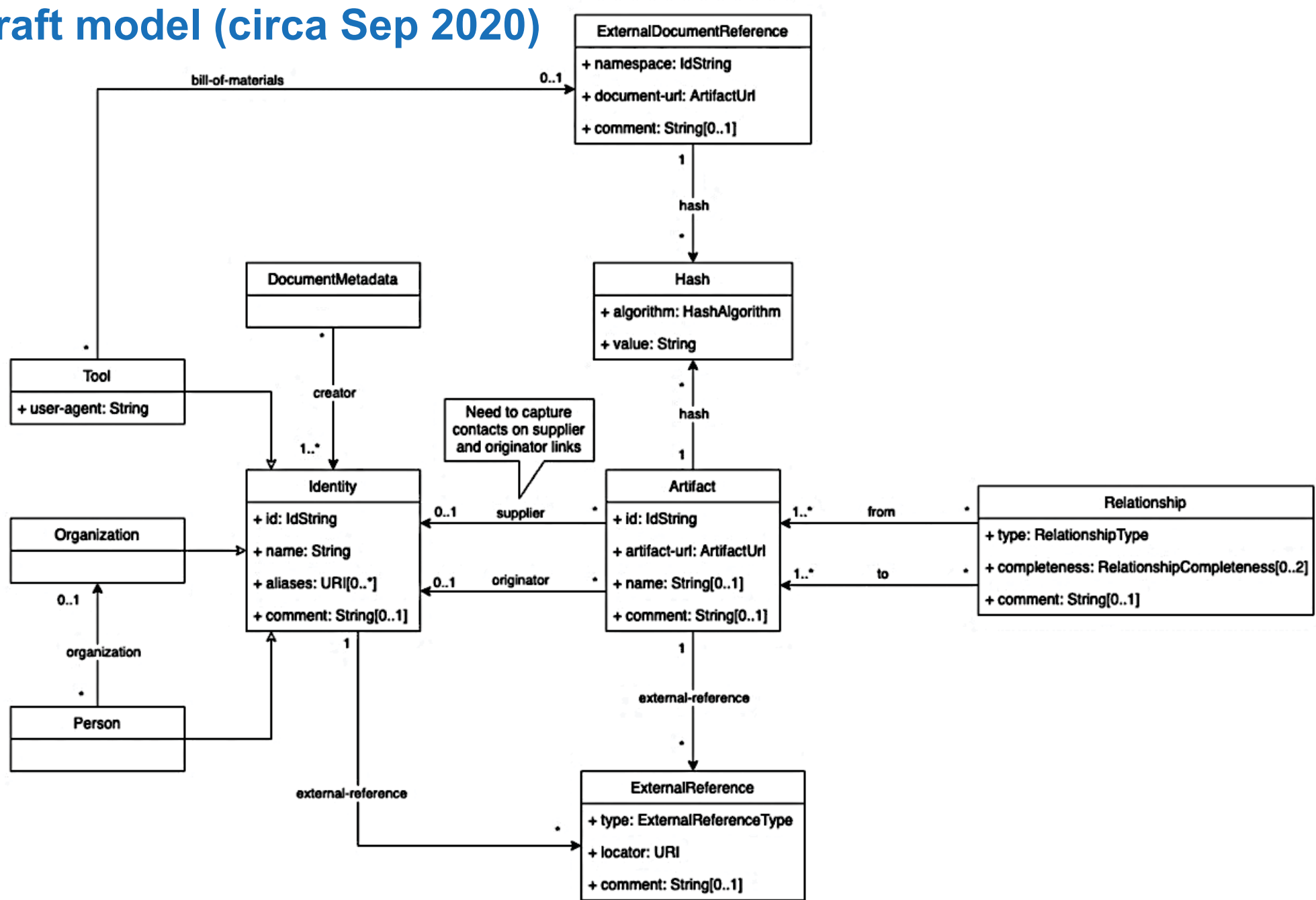
<https://www.mitre.org/publications/technical-papers/standardizing-sbom-within-the-sw-development-tooling-ecosystem>

3T-SBOM Standard Timeline

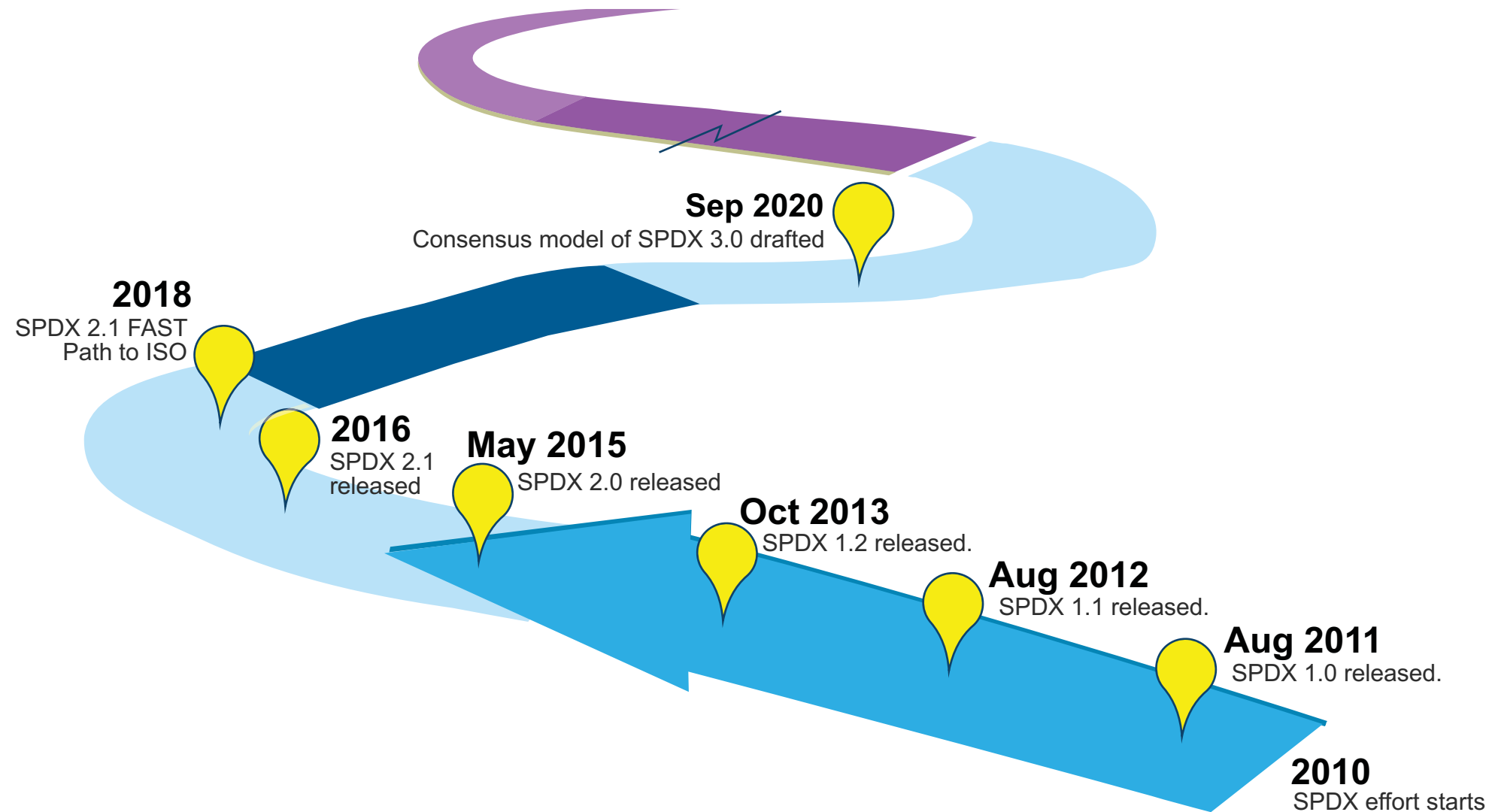


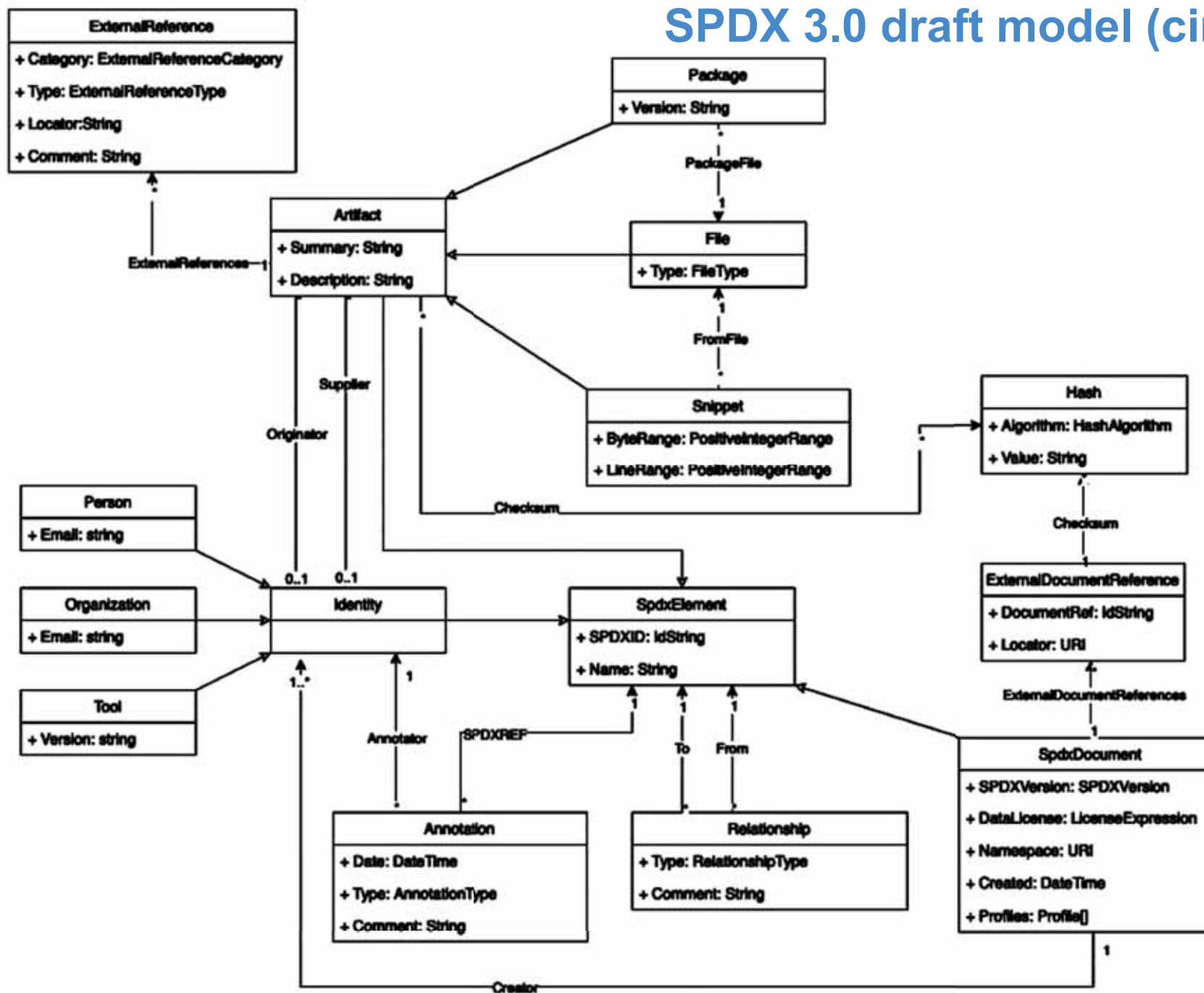
CISQ/OMG Tool-to-Tool Software Bill of Materials Effort

3T-SBOM draft model (circa Sep 2020)



SPDX SBOM Standard Timeline





ExternalReference

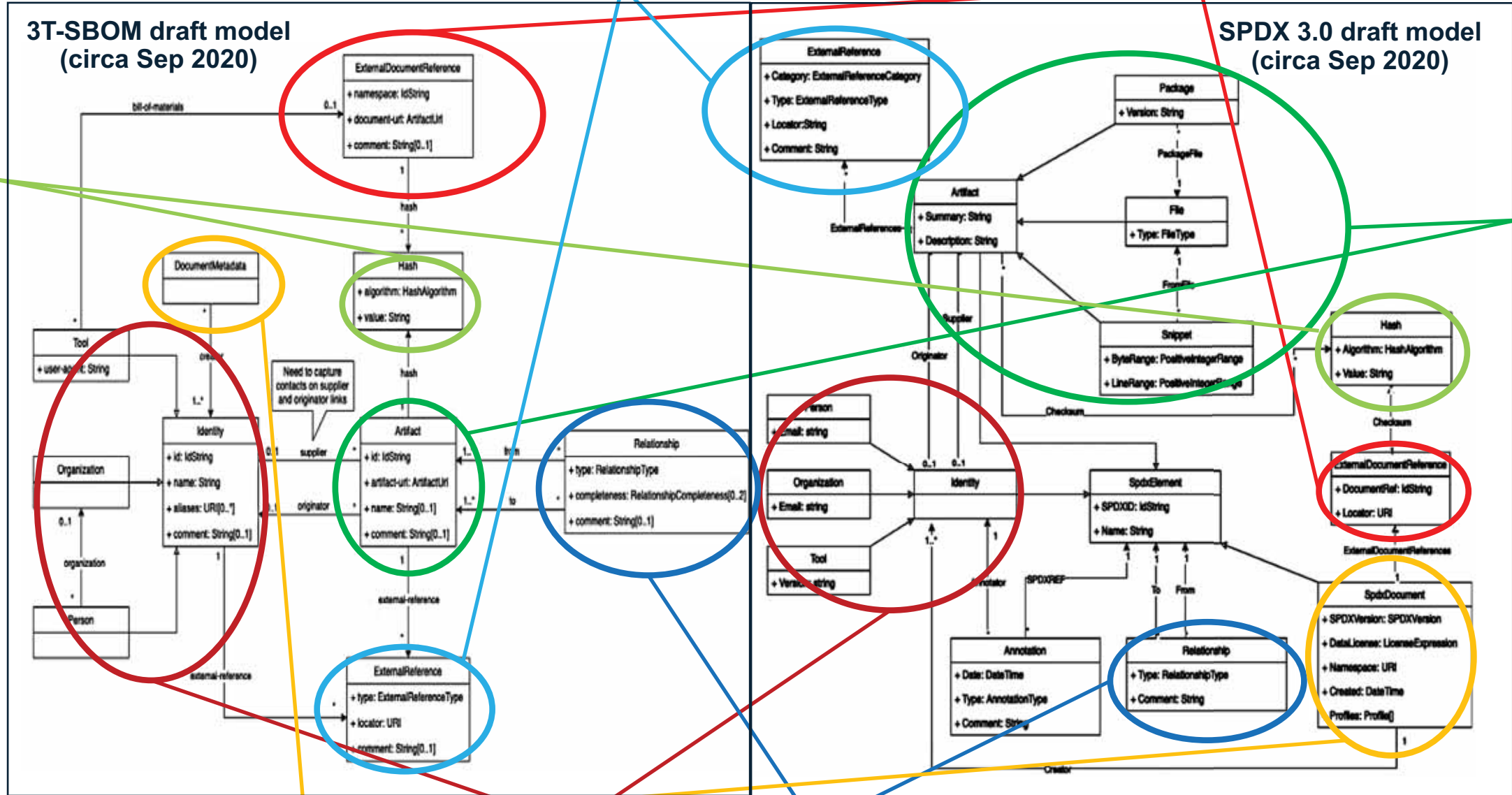
ExternalDocumentReference

3T-SBOM draft model (circa Sep 2020)

SPDX 3.0 draft model (circa Sep 2020)

Hash

Artifact



DocumentMetadata

Identity

Relationship

Who is Using SPDX?

- Wide range of users - from sophisticated security software analysts to non-technical procurement and compliance executives
 - Requires a wide range of “serialization” formats ranging from RDF/XML, to JSON, to YAML, to Spreadsheets
- Software and Software as a Service providers - from the very large to to the very small including many software tooling vendors
 - Many providing or utilizing open source libraries which support overall adoption
- Open source package maintainers across a wide variety of ecosystems (e.g. Python Poetry, Maven, Gradle, NPM)

Jan 2021: Merged w/SPDX

Common goals

Already working together

Fewer meetings



SPDX Governance

release 3.0 will be reviewed by the OMG Architecture Board

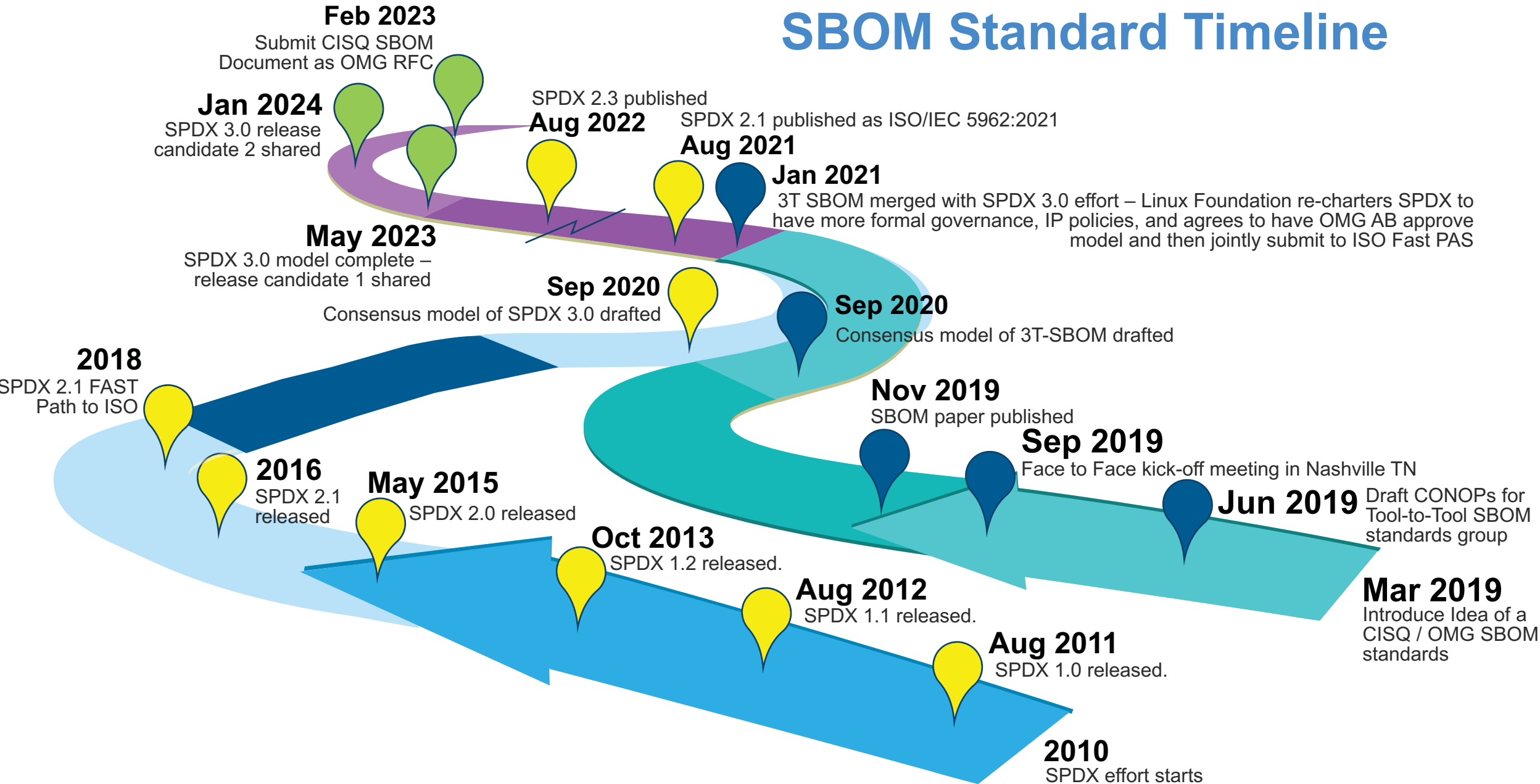
Working group

Technical Team

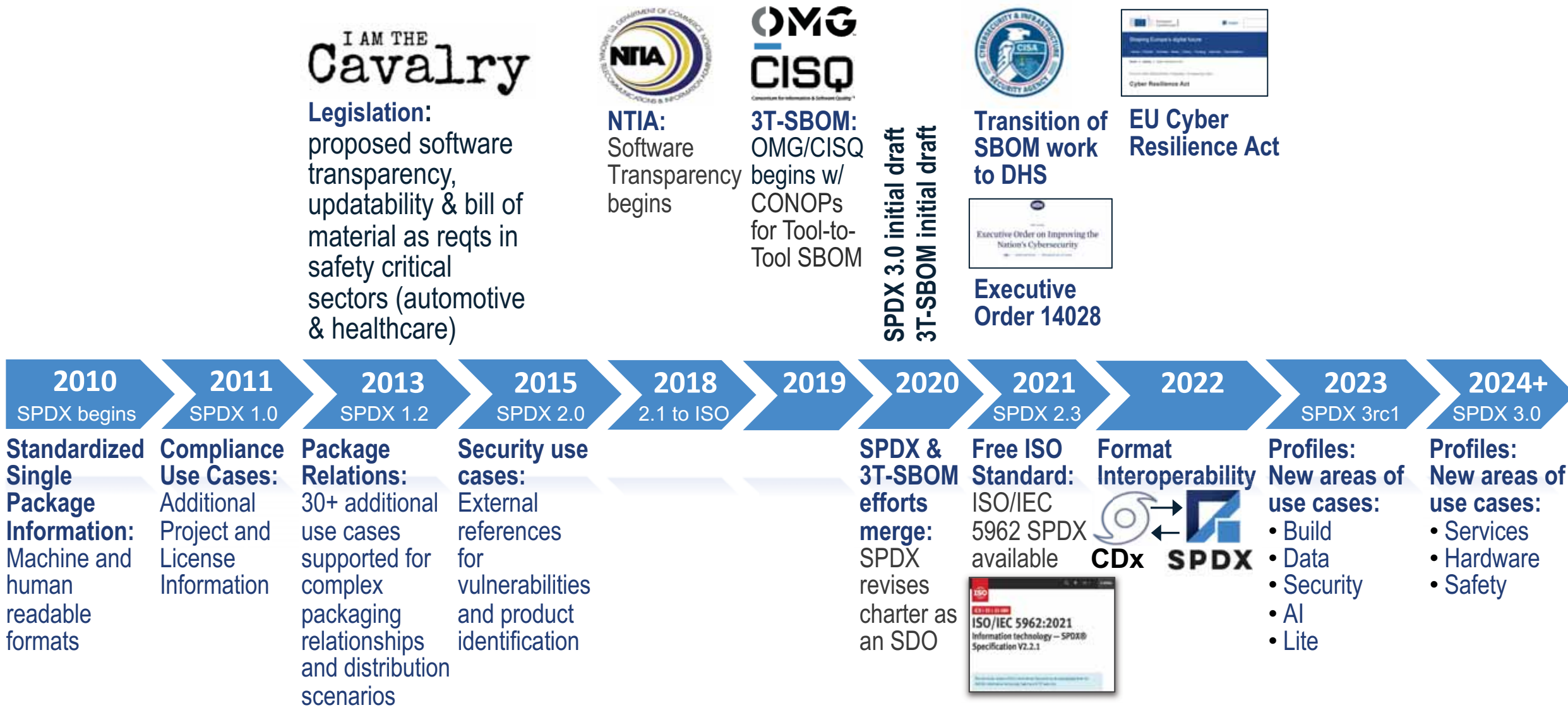
Steering Committee

- Open
 - Is open to contributions from anyone interested - all team and working group meetings are open
 - Team leads nominated by any participant and steering committee composed of team leads
- Transparent
 - Most of our work is maintained in the SPDX Github repo including minutes
- Inclusive
 - Leads and steering committee consists of individuals from diverse backgrounds and industries
 - Over 40 organizations have contributed directly to the SPDX spec representing a majority of industry segments (Technology, Critical Infrastructure, Financial, Gov't) and geographies

SBOM Standard Timeline



Timeline of SPDX Evolution - Use Case by Use Case



THE LINUX FOUNDATION PROJECT


SPDX ABOUT - LEARN - ENGAGE - USE - NEWS [JOIN US](#)

The Software Package Data Exchange (SPDX)

An open standard describing SBOMs (Software Bill of Materials), communicating a release name, version, components, licenses, copyrights, and useful security references. As a common format, SPDX reduces redundant work related to sharing important release data, thereby streamlining distribution and compliance.


The SPDX specification is a freely available international open standard (ISO/IEC 5962:2021).

[LEARN MORE](#)




Learn
Learn more about the structure of SPDX and how to participate.

[ABOUT SPDX](#)



Use
Explore the ways that you can engage with SPDX.

[USE SPOX](#)



Tools
SPDX workgroup tools and others you can use.

[SPDX TOOLS](#)

Areas of Interest

SPDX is organized in areas of interest or profiles focused on specific user needs.

Supported by These Foundations

Latest SPDX News

NOV 8, 2023 BLOG

Capturing Software Vulnerability Data in SPDX 3.0

The flexibility of SPDX 3.0 allows users to either link SBOMs to external security vulnerability data or to embed security vulnerability information in the SPDX 3.0 data format, thanks to support for a security-specific profile. This is different from SPDX version 2, which enabled users to link an SBOM to...

[READ MORE](#)














































OCT 9, 2023 BLOG

Understanding SPDX Profiles

On the surface, profiles are pretty straight forward - they are a way of organizing a specification that covers a broad array of use cases into "profiles" more specific to what a specific producer or consumer of SPDX data may be interested in.

[READ MORE](#)


SPDX Supporters

<https://spdx.dev/>

















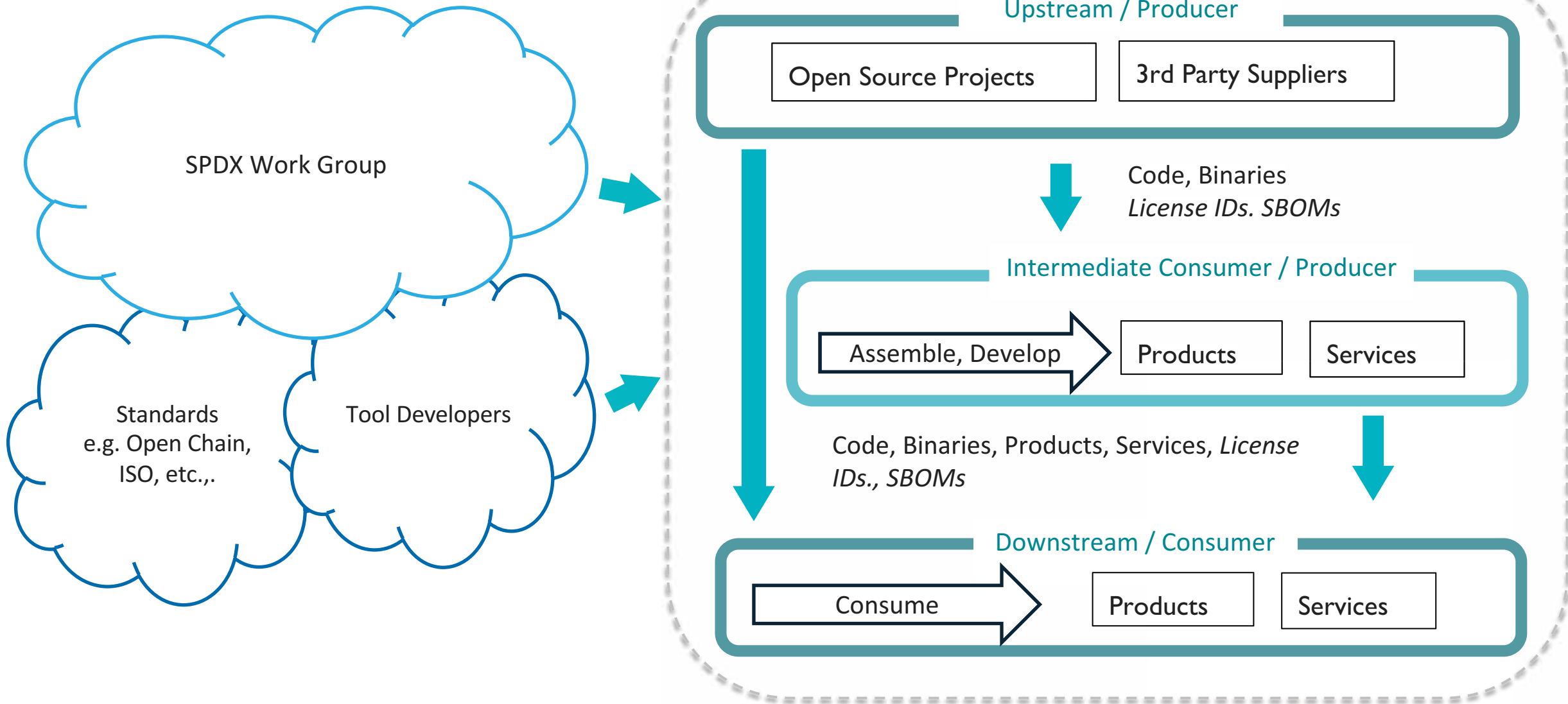







Copyright © 2023 The Linux Foundation®. All rights reserved. The Linux Foundation has registered trademarks and uses trademarks. For a list of trademarks of The Linux Foundation, please see our Trademark Usage page. Linux is a registered trademark of Linux Torvalds. Privacy Policy and Terms of Use.

SPDX SBOM Ecosystem



SPDX 3 model

The Software Package Data Exchange® (SPDX®) is a standard format for communicating information about components associated with software packages. It has wide industry adoption as a standardized Software Bill

☰ README.md

This repository holds the model for the information captured on the (upcoming) SPDX version 3 standard.

Branches and Formats

The editable files are written in a constrained subset of Markdown and are stored in the `main` branch.

These files are automatically processed into the following formats available at <https://spdx.github.io/spdx-3-model/>:

- HTML files generated by [Ontospy](https://spdx.github.io/spdx-3-model/auto-generated/): <https://spdx.github.io/spdx-3-model/auto-generated/>
- [JSON-LD context](#) file: [context.json](#)
- Model [SHACL](#) and [OWL](#) files:
 - [Turtle format](#): [model.ttl](#)
 - [JSON-LD format](#): [model.jsonld](#)

Note that these files are also available in the `gh-pages` branch.

People who wish to read the current version of the information should be viewing the generated files, while anyone wanting to edit should be working on the former.

For information about how to contribute to a specific profile, please see [Contributing.md](#).

Contribute!

Feel free to join us and contribute! The discussions are happening on the `spdx-tech` mailing list and during our weekly meetings. All the details are in: <https://spdx.dev/participate/tech/>

Product Solutions Open Source Pricing Search or jump to... Sign in Sign up

spdx / spdx-3-model Public Notifications Fork 37 Star 40

Code Issues 93 Pull requests 18 Actions Projects Security Insights

main 40 branches 1 tag Go to file Code About

rjudge and goneall Consistent naming for packageUri 96cb42a 3 days ago 482 commits

.github/workflows	add another check that validates that the context for jsonld	last month
model	Consistent naming for packageUri	3 days ago
serialization	Fix context URLs in JSON-LD docs	last week
.gitignore	WIP: initial draft of Licensing content	last year
CLA.md	added licensing information	2 years ago
Contributing.md	Update Contributing.md (#337)	7 months ago
Glossary.md	Update Glossary with proposed profile definition	3 months ago
Governance.md	Remove duplicate governance docs (#294)	7 months ago
License.md	added licensing information	2 years ago
README.md	Update README to the current state of generated files	3 months ago
model-security-profile.drawio	updates 3.0 security model diagram for consistency with 3.0 core ...	3 months ago
model-security-profile.png	updates 3.0 model-security-profile.png	3 months ago
model.drawio	Changed name X-Collection to SpdxDocument aligned with Tech c...	last week
model.png	Changed name X-Collection to SpdxDocument aligned with Tech c...	last week

No description, website, or topics provided.

Readme View license Activity 40 stars 18 watching 37 forks Report repository

Releases 1 tags

Packages No packages published

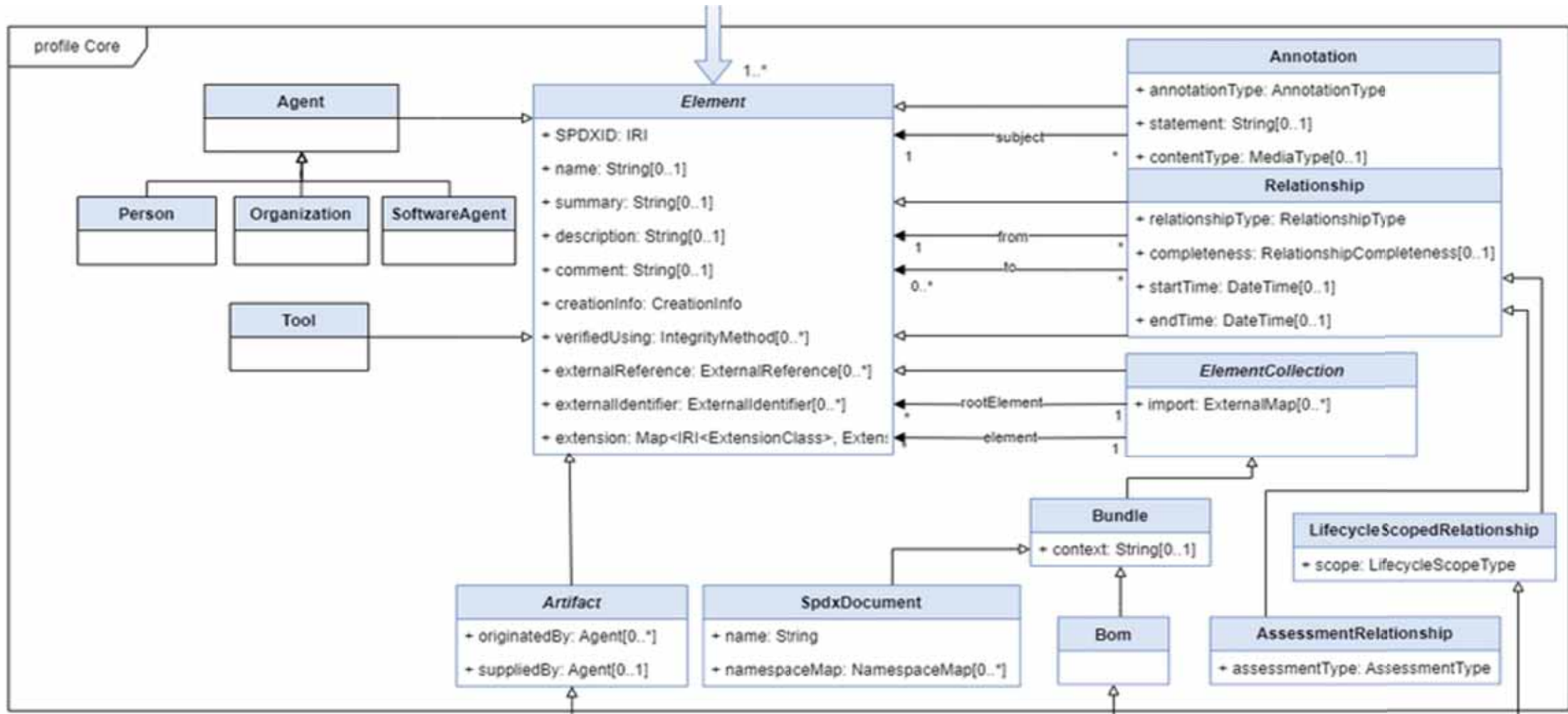
Contributors 25 + 11 contributors

<https://github.com/spdx/spdx-3-model>

Core Profile Overview



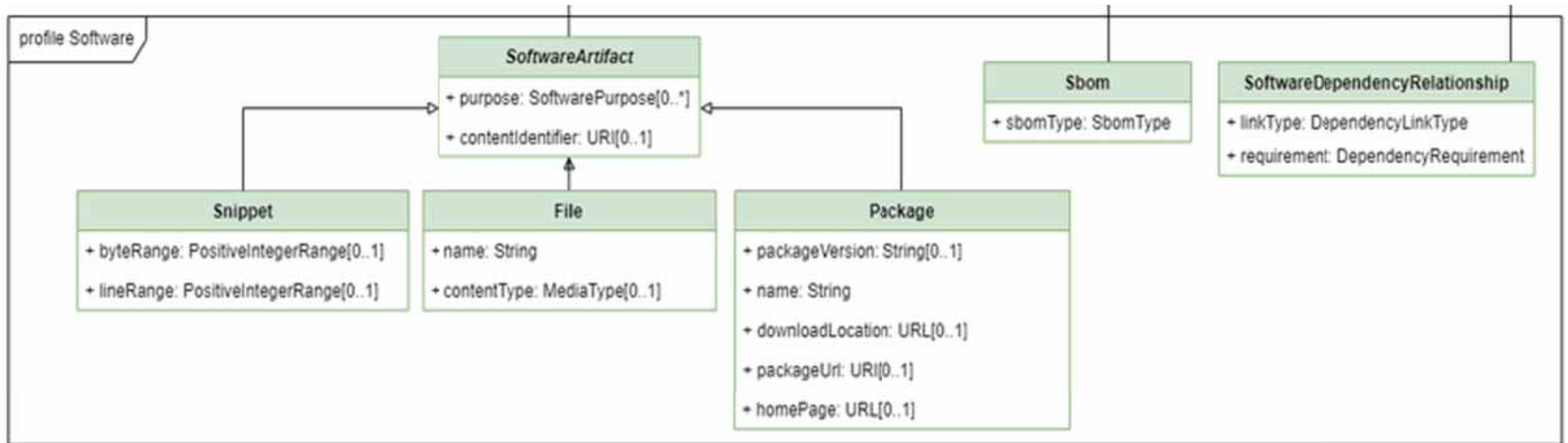
- Defines foundational concepts which are the basis for all SPDX 3.0 profiles



Software Profile Overview



- Defines concepts related to software artifacts
- Introduces and supports CISA SBOM Types
 - Design/Source/Build/Deployed/Runtime/Analyzed



<https://www.cisa.gov/sites/default/files/2023-04/sbom-types-document-508c.pdf>

SPDX 3.0 Simple Data Types, Complex Data Types and Enumerations

Simple Data Types

SemVer: String	<h2>Legend</h2> <p><i>Italics</i> - abstract, you must use a subclass</p> <p><u>Underscore</u> - value type/struct semantics, equality determined by comparing values</p>
String constrained to SemVer 2.0.0 specification.	
MediaType: String	
String constrained to RFC 2046 specification.	
EmailAddress: String	
String constrained to RFC 2822 specification.	

Complex Data Types

These types have value type/struct semantics - equality is determined by comparing values and they MUST NOT be referenced by name across documents. Serialization formats MAY enable de-duplication within a single document.

CreationInfo + specVersion: SemVer + profile: ProfileIdentifier[1..*] + created: DateTime + dataLicense: 'CC0' + createdBy: Agent[1..*] + createdUsing: Tool[0..*] + comment: String[0..1]	ExternalIdentifier + externalIdentifierType: ExternalIdentifierType + identifier: String + comment: String[0..1]	PositiveIntegerRange + start: PositiveInteger + end: PositiveInteger
NamespaceMap + prefix: String + namespace: IRI	ExternalReference + externalReferenceType: ExternalReferenceType + locator: IRI + contentType: MediaType[0..1] + comment: String[0..1]	ProfileIdentifier + name: String
ExternalMap + externalId: IRI + locationHint: URL[0..1] + verifiedUsing: IntegrityMethod[0..*] + definingArtifact: Artifact	IntegrityMethod + comment: String[0..1]	ExtensionClass
	Hash + algorithm: HashAlgorithm + hashValue: byte[1..*]	

Enumerations

RelationshipType Meta describes [bundle->artifact] amends [element->element] other [element->element] (comment) Structure contains [artifact->artifact] Behavioral dependsOn [artifact->artifact] patches [artifact->artifact] tests [artifact->artifact] Pedigree generates [artifact->artifact] expandedFromArchive fileAdded fileDeleted fileModified copy [artifact->artifact] packages (obsolete?) Provenance ancestor [artifact->artifact] availableFrom [artifact->identity] variant [artifact->artifact] Serialization hasSerialization [SpdxDocument ->artifact] Obsolete? buildTool devTool testTool dependencyManifest distributionArtifact example dataFile testCase documentation metafile test requirementFor specificationFor	ExternalReferenceType altDownloadLocation altWebPage securityAdvisory securityFix securityOther other	SoftwarePurpose application archive bom configuration container data device documentation executable file firmware framework install library module operatingSystem patch source other	AssessmentType cvss2 cvss3 epss exploitCatalog vex ssvc other
RelationshipCompleteness complete [Default] incomplete noAssertion	HashAlgorithm sha1 sha224 sha256 [default] sha384 sha512 sha3_224 sha3_256 sha3_384 sha3_512 md2 md4 md5 md6 spdx_pvc_sha1 spdx_pvc_sha256 blake2b_256 blake2b_384 blake2b_512 blake3 other	DependencyLinkType noAssertion [default] static dynamic tool other	
	ExternalIdentifierType cpe_2.2 cpe_2.3 email purl uri-scheme swid [deprecate?] swid [deprecate?] gitoid [deprecate?] other	DependencyScope noAssertion [default] build dev test runtime other	DependencyRequirement noAssertion [default] optional required provided preRequisite
	AnnotationType review other	SbomType TBD	

SPDX Profiles Overview



Security information - vulnerability details related to software



Build related information - provenance and reproducible builds



Information about AI models - ethical, security, and model data



Information about datasets - AI and other data use cases



Information about copyrights and licenses - supports compliance

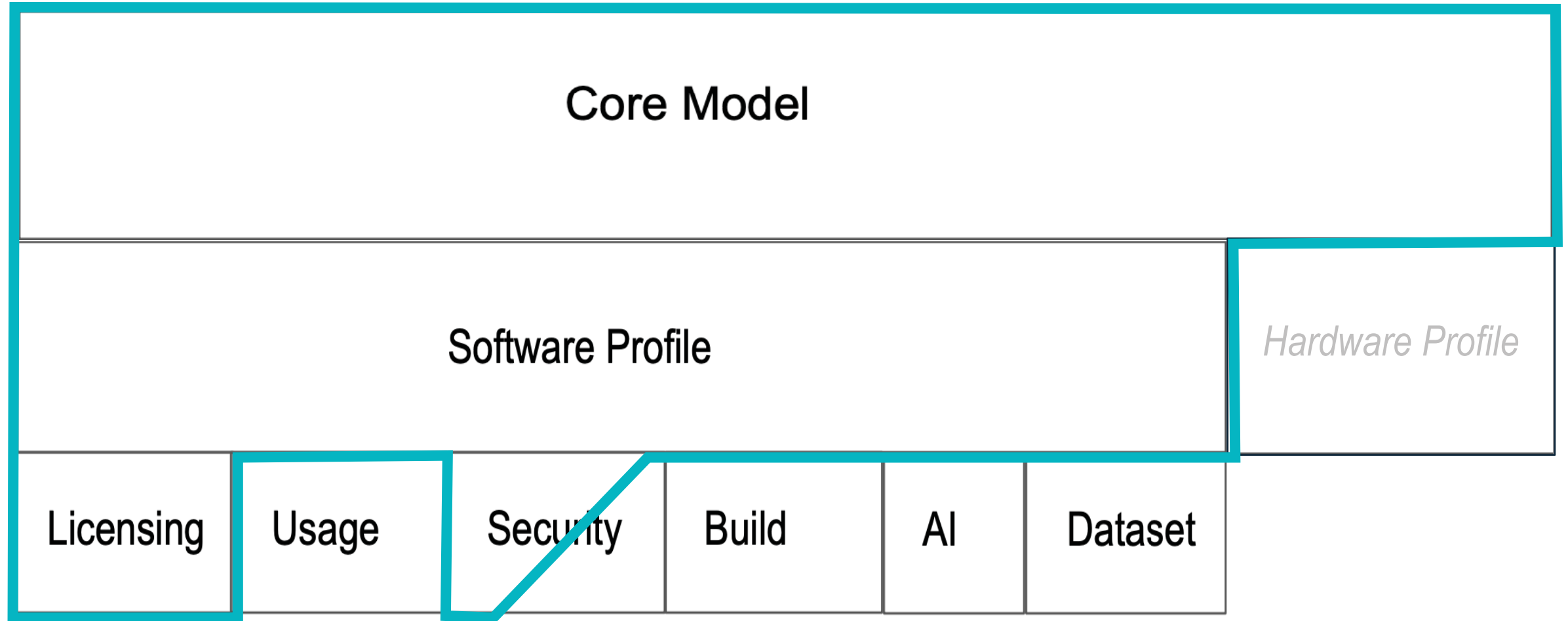


Information specific to software



Information used across all profiles

SPDX 3.0 Increases Modularity



SPDX 3.0 (Core Model + Software Profile + ~Security Profile + Licensing Profile) == SPDX 2.3

Security Profile Overview



Captures security-related metadata specific to a piece of software

- Security profile v3.0 properties and relationships support vulnerability management.
 - **discovery & disclosure:** Communicate the vulnerabilities found by person (auditor/researcher), tool, or organization in a particular piece of software, e.g. CVE
 - **severity:** Communicate severity and rank vulnerabilities in a specific piece of software using industry standardized methods, e.g. CVSS
 - **risk:** Communicate how a vulnerability affects a specific piece of software, e.g., EPSS, VEX
 - **remediation:** Communicate how a vulnerability may be addressed or has already been addressed for a particular piece of software
- Security assurances for software integrity and other secure SDL activities are also supported in SPDX 3.0

Build Profile Overview



Use Cases

Build Provenance

Providing and verifying build provenance of a software ensure that the provided software is what it claims to be

Security

Providing the necessary information to respond to supply chain compromises such as a bad builder or build tool/configuration

Audit

Checking if certain build tool (e.g. security compilation flag configuration) is enabled in the creation of the software

Reproducibility

Providing the necessary information to verify the reproducibility of a build to create more confidence in the built artifact and the build process.

Quality

Provide evidence to assert higher confidence that the specified software components in an SBOM is accurate and complete.

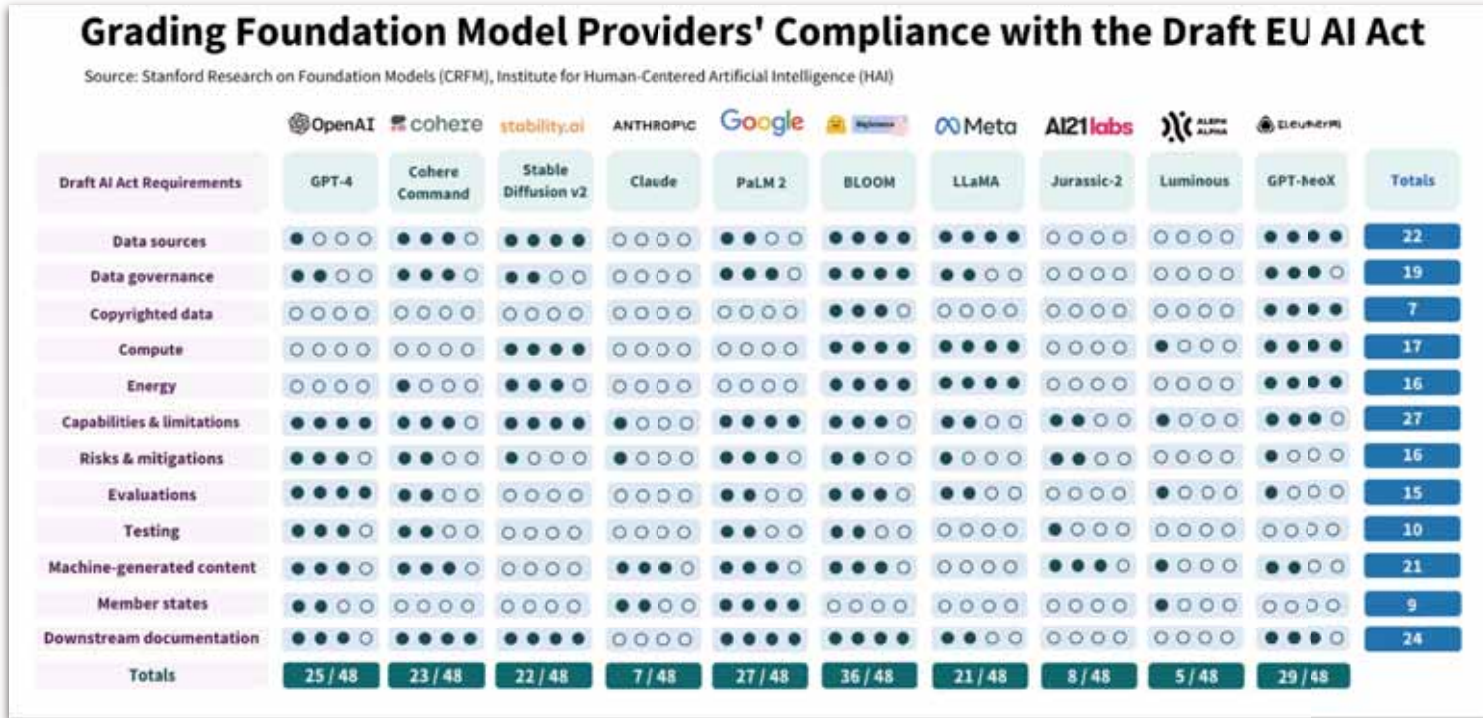
Safety

Providing information about all the inputs and processes that are used in the creation of the software so they can be audited and verified for safety standards in critical infrastructure and components.

AI Profile Overview



Automating Issue Identification Requires Machine Readable AI BOM



- Many issues exist even when comparing to one standard (EU AI Act) across most of the existing state-of-the-art AI models
- In order to be able to establish compliance to several regulations around the world and identify potential risks associated with the use and adaption of AI software an automated standard, that captures all the required data and is machine readable is pivotal.

Data Profile Overview



A profile that adds on top of the core-software profile to describe the dataset that is used to train or test an AI software. These datasets could also be used for other purposes. Similar, to AI profile, we take special care to ensure that process of forming a dataset is captured. In addition, we also make it a point to capture the provenance and the lineage associated with the dataset. We introduce new fields to do so as provenance and lineage of a dataset has more facets to it compared to traditional software.

Properties borrowed from other profiles

- originatedBy
- downloadLocation
- builtTime
- primaryPurpose
- releaseTime

Properties specific to Dataset profile

- datasetType
- datasetCollectionProcess
- intendedUse
- datasetSize
- datasetNoise
- datasetSize
- dataPreprocessing
- sensor
- knownBias
- sensitivePersonalInformation
- anonymizationMethodUsed
- confidentialityLevel
- datasetUpdateMechanism
- datasetAvailability

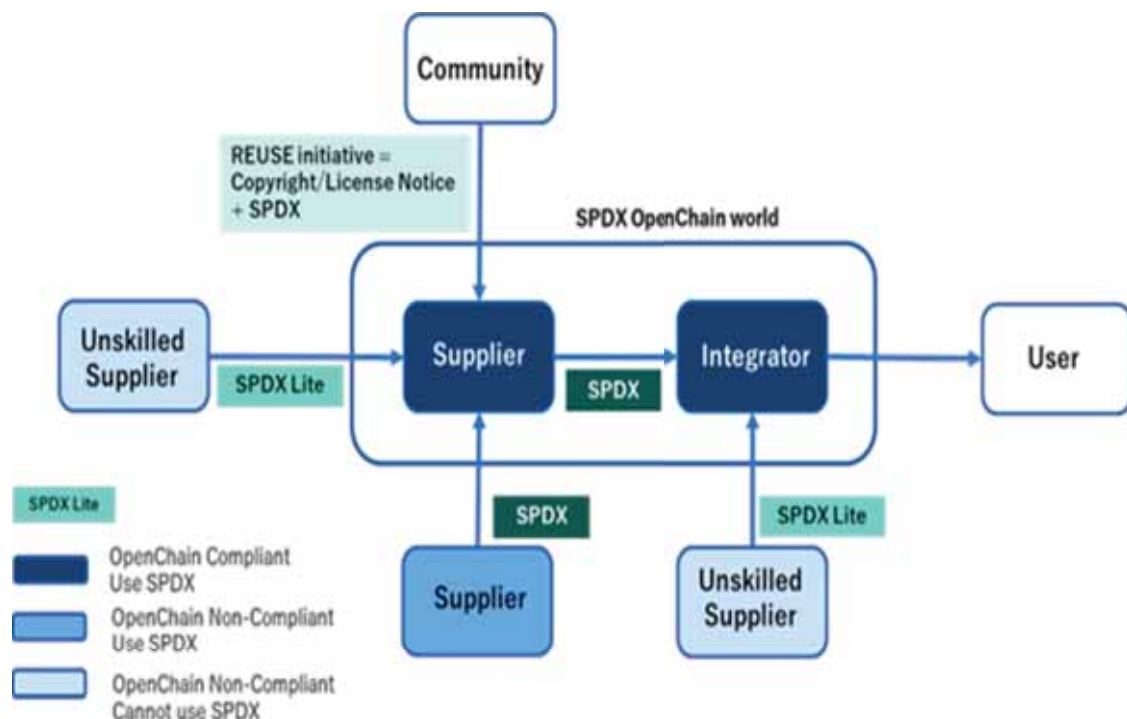
Licensing Profile Overview



- Classes for License model components
 - includes those previously defined in SPDX 2.3 (and earlier) RDF, as well as SPDX License List fields
 - represents Licenses, License Exceptions, and the data model for license combinations (AND, OR, WITH, +)
 - now with License Additions (custom text for right side of WITH expressions)
- Relationships for Licensing-related metadata
 - conceptually similar to SPDX 2.3
 - better clarity and alignment across Software Artifact types
 - now structured as SPDX Relationships rather than properties
 - Enables one SPDX data creator to define the Software Artifacts, and others to later specify their conclusions as to licenses by reference to the same Software Artifact SPDX elements

SPDX Lite Overview

A subset spec. of SPDX focuses on the **minimum elements for license compliance**. The design policy provides clear information for easy practice, intended for use in the **ISO/IEC 5230** and **PRF 18974** processes.

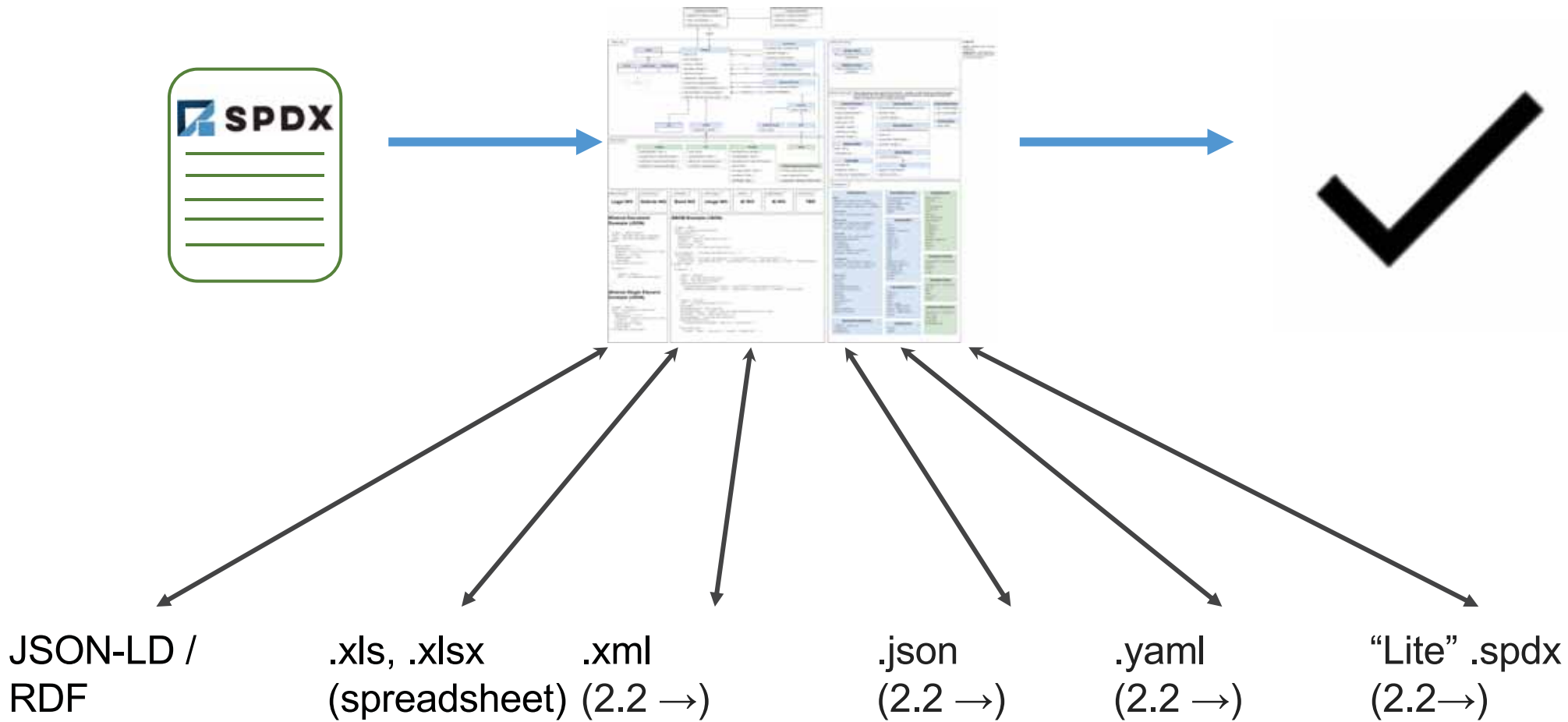


Package-centric

#	SPDX subclass	Field name
L1.1	6.1	SPDX Version
L1.2	6.2	Data License
L1.3	6.3	SPDX Identifier
L1.4	6.4	Document Name
L1.5	6.5	SPDX Document Namespace
L1.6	6.8	Creator
L1.7	6.9	Created
L2.1	7.1	Package Name
L2.2	7.2	Package SPDX Identifier
L2.3	7.3	Package Version
L2.4	7.4	Package File Name
L2.5	7.5	Package Supplier
L2.6	7.7	Package Download Location

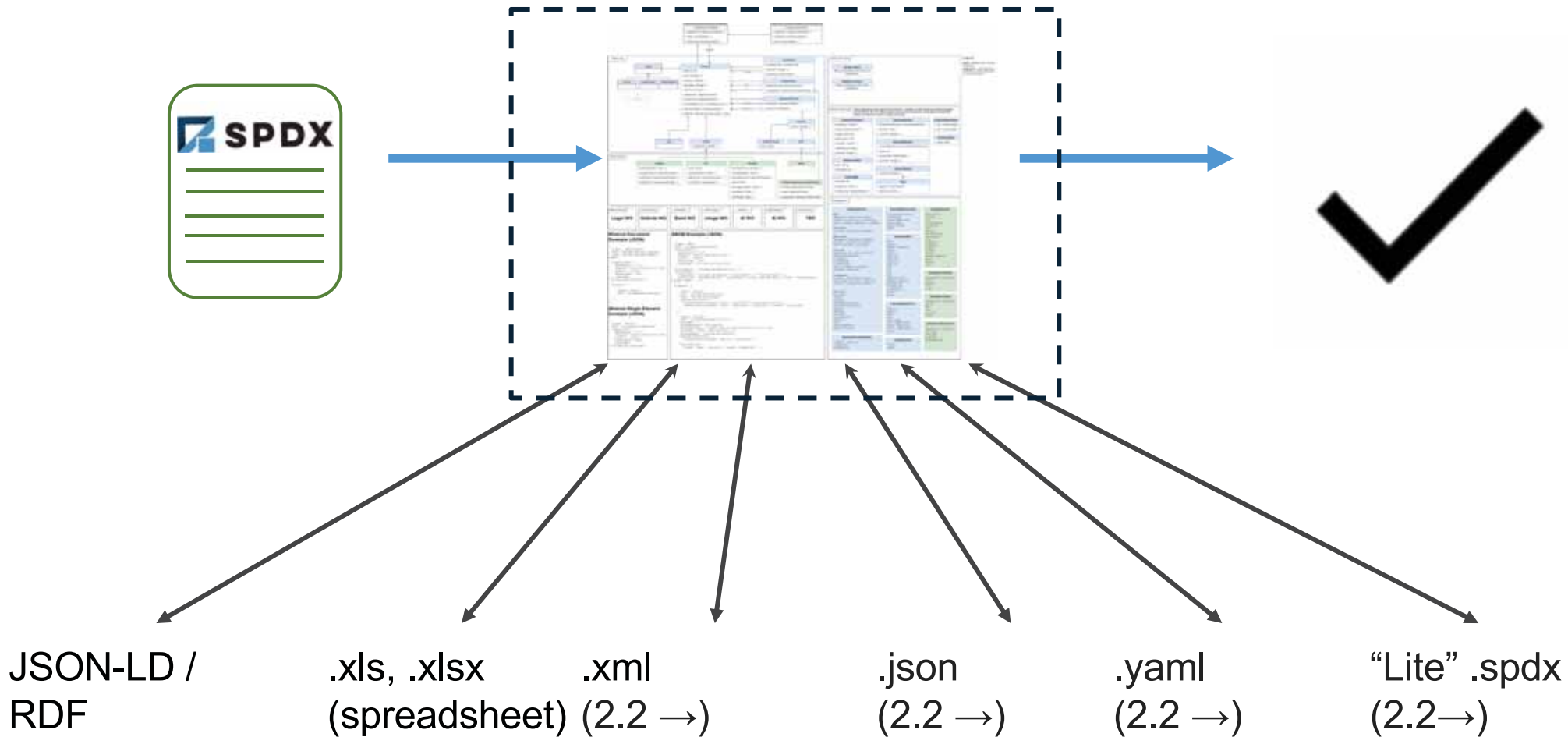
#	SPDX subclass	Field name
L2.7	7.8	Files Analyzed
L2.8	7.11	Package Home Page
L2.9	7.13	Concluded License
L2.10	7.15	Declared License
L2.11	7.16	Comments on License
L2.12	7.17	Copyright Text
L2.13	7.20	Package Comment
L3.1	10.1	License Identifier
L3.2	10.2	Extracted Text
L3.3	10.3	License Name
L3.4	10.5	License Comment

Where is the SPDX 3.0 development?

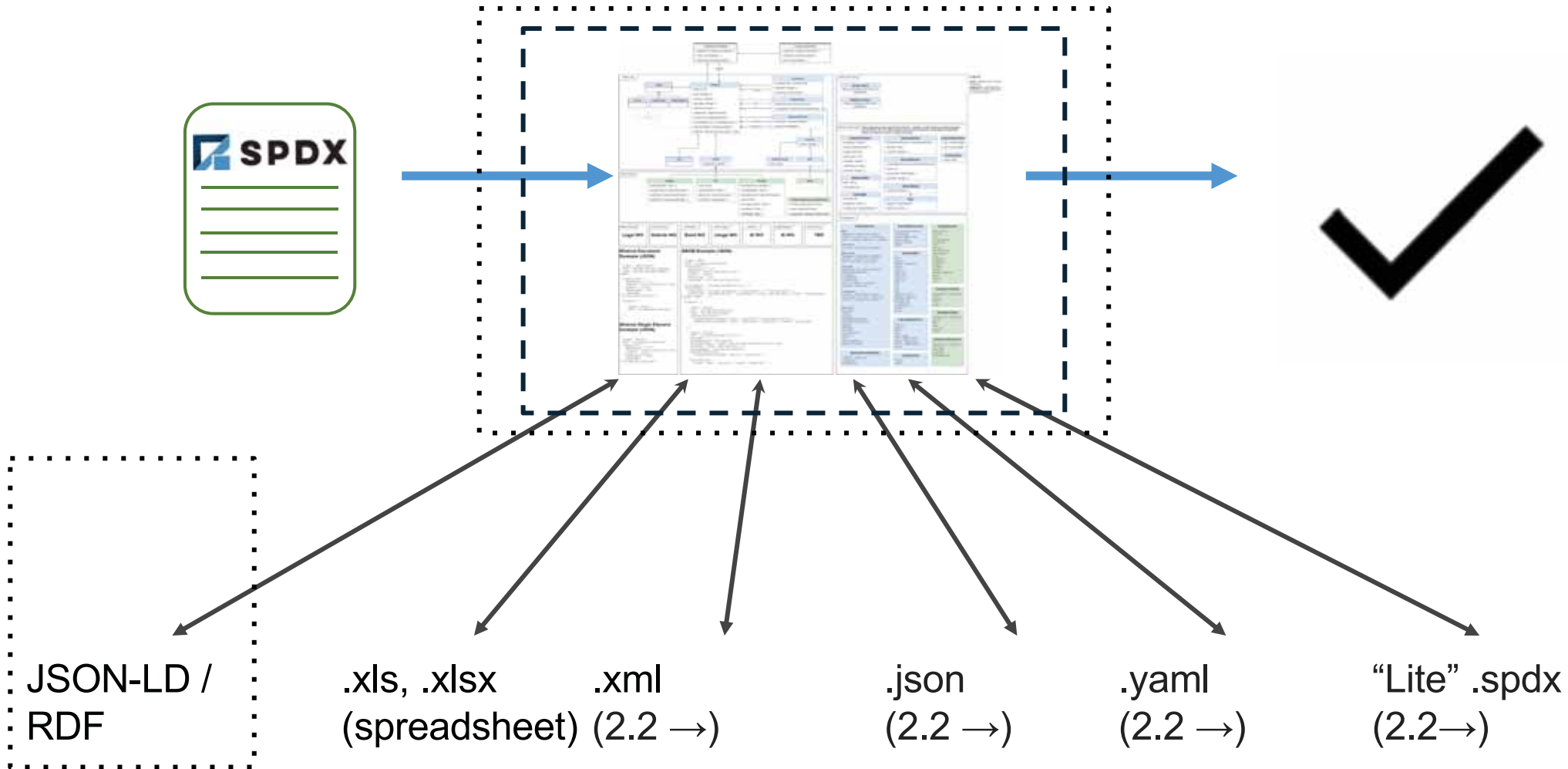


Where is the SPDX 3.0 development?

RC1



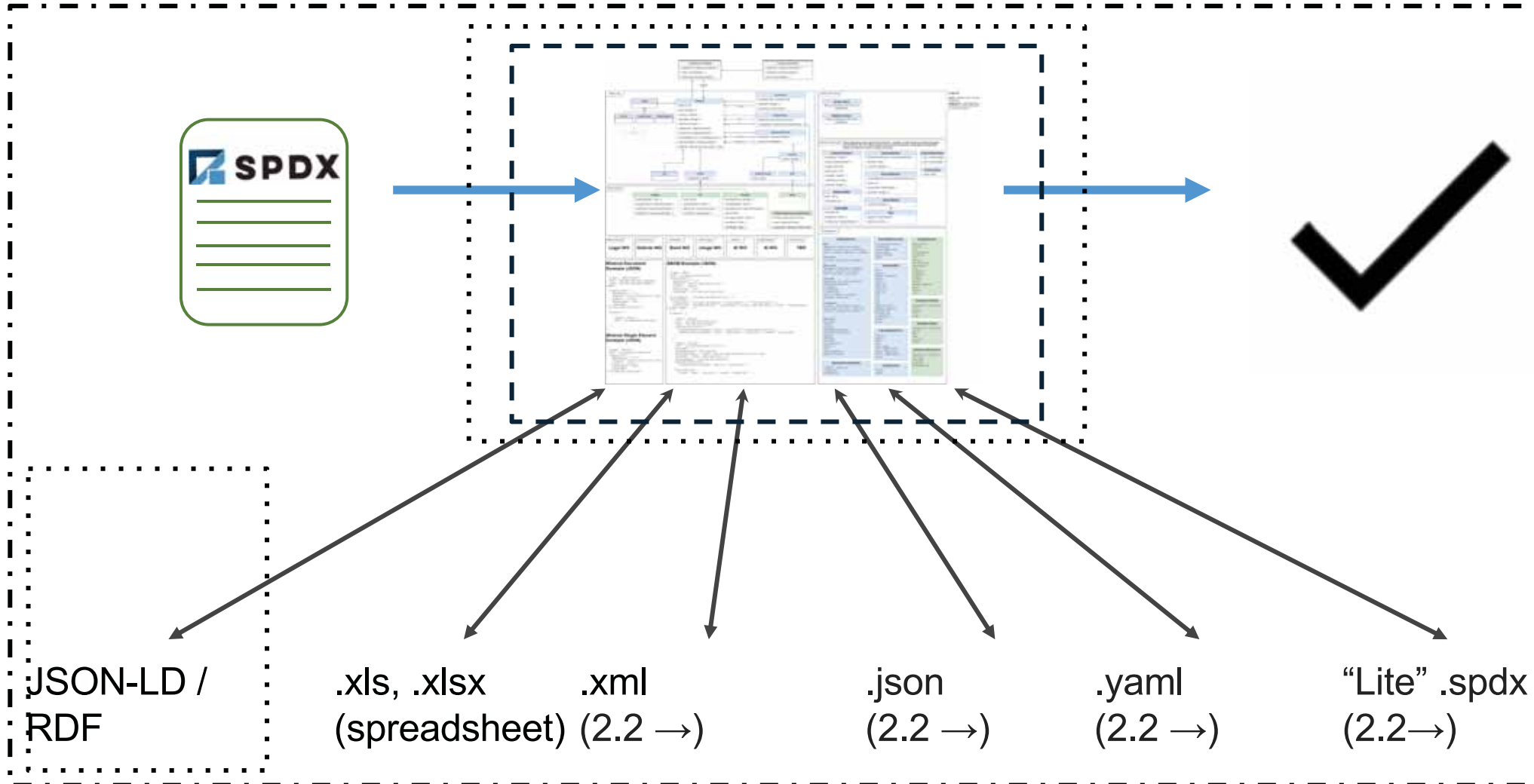
Where is the SPDX 3.0 development?



RC1

RC2

Where is the SPDX 3.0 development?






RC1

RC2

Release

Beyond SPDX 3.0

- Software as a Service 
 - Profile team has been active for about 3 months
 - Tracking the CISA workgroup with participation from Nisha
- Safety 
 - We're learning that we already have most of what we need in 3.0
 - For more info see <https://www.linux.com/featured/sboms-supporting-safety-critical-software/>
- Hardware 
 - Safety Standards expect to know “system” that software is running on
 - Potential participants include RISC-V & ARM core adopters, Chips Alliance Members and Board Manufacturers

SPDX 3.0 Proposed Rollout

Jan/Feb - Profiles added to Model will be considered part of 3.0 ready to be reviewed

- Minimum will have Security, Build & Licensing (equivalence to 2.3)
- Serializations in JSON prototyped in Java & Python libraries
- Need to identify leads for JSON, YAML, RDF, Tag:Value, XML

March - Additional serializations added to libraries.

- Porting guide from 2.2/2.3 → 3.0 drafted, and available for review
- Provide examples of how to use new SPDX 3.0 features.
- Release candidate specification for SPDX 3.0 for other tools feedback

April/December - Release SPDX 3.0 release candidates

- Gather community feedback, and improve documentation

February - bring to OMG Architecture Board as an RFC once ready to start ISO submission update

- if no significant errata after 2 months, start work on updating ISO submission

How to Get Involved - PRs & Issues

- How to Get Involved - PRs & Issues
 - <https://github.com/spdx>
- Quick start guides
 - <https://github.com/spdx/outreach/tree/main/quickstart>
- Specification
 - <https://github.com/spdx/spdx-spec> ← ISO submission format
 - <https://github.com/spdx/spdx-3-model> ← 3.0 development
 - <https://github.com/spdx/spdx-examples>
- Tooling
 - <https://github.com/spdx/tools-python>
 - <https://github.com/spdx/tools-golang>
 - <https://github.com/spdx/tools-java>
- License List
 - <https://github.com/spdx/license-list-XML>

Questions?