CycloneDX 1.3				CycloneDX 1.4				SPDX 2.2				
Class / Type	Property	Property Type	Cardinality	Class / Type	Property	Property Type	Cardinality	Class / Type	Property	Property Type	Cardinality	Notes
[Top Level]	bomFormat	String - "CycloneDX"	1..1	[Top Level]	bomFormat	String - "CycloneDX"	1..1					Fixed string - not applicable
[Top Level]	specVersion	String	1..1	[Top Level]	specVersion	String	1..1	[Top Level]	spdxVersion	String	1..1	
[Top Level]	serialNumber	String	0..1	[Top Level]	serialNumber	String	0..1	[Top Level]	documentNamespace	String - URI	1..1	Converted to a URI if present, otherwise a new namespace is created with a random UUID
[Top Level]	version	integer	1..1	[Top Level]	version	integer	1..1	[Top Level]	documentNamespace	String - URI	1..1	Version is appended to the SPDX document namespace
[Top Level]	metadata	Metadata Object	0..1	[Top Level]	metadata	Metadata Object	0..1	[Top Level]	creationInfo	CreationInfo	1..1	
[Top Level]	components	Component Object	0..*	[Top Level]	components	Component Object	0..*	[Top Level]	packages, files	Package Object	0..*	All components are added. If there is no metadata correlated to the SPDX document describes, all of these components are added as the document describes elements
[Top Level]	services	Service Object	0..*	[Top Level]	services	Service Object	0..*	[Top Level]	annotations	Annotation Object	0..*	No SPDX equiv.
[Top Level]	externalReferences	ExteranlReference Object	0..*	[Top Level]	externalReferences	ExteranlReference Object	0..*	[Top Level]	annotations	Annotation Object	0..*	Not applicable for the SPDX document level - only at the component level. One possibility is to create an SPDX package for the SBOM which would contain this information
[Top Level]	dependencies	Dependency Object	0..*	[Top Level]	dependencies	Dependency Object	0..*	[Top Level]	relationships	Relationship	0..*	Relationship type dependency of
[Top Level]	compositions	Composition Object	0..*	[Top Level]	compositions	Composition Object	0..*	[Top Level]	relationships	Relationship	0..*	Completeness information is added as comments to the Relationship.
				[Top Level]	vulnerabilities	Vulnerability Object	0..*					No SPDX equiv. - plan to generate separate VEX document
				[Top Level]	signature	Signature Object	0..1					No SPDX equiv. - plan to generate separate VEX document
								[Top Level]	SPDXID	String	1..1	Constant - always the same value
								[Top Level]	reviewed	Review Object	0..*	Deprecated as of 2.0
								[Top Level]	hasExtractedLiceningInfos	ExtractedLicense Object	0..*	Currently not used - this could be used to capture license text from the license object in the future
								[Top Level]	name	String	1..1	Generated from the documentDescribes
								[Top Level]	comment	String	0..1	Not used
								[Top Level]	snippets	Snippet Object	0..*	No equivalent in CDX
MetaData	timestamp	String	0..1	MetaData	timestamp	String	0..1	CreationInfo	created	Timestamp	1..1	
MetaData	tools	Tool	0..*	MetaData	tools	Tool	0..*	CreationInfo	creators	String	1..*	Checksums are captured in Annotations
MetaData	author	OrganizationalContact	0..*	MetaData	author	OrganizationalContact	0..*	CreationInfo	creators	String	1..*	Contact phone numbers and multiple emails care captured in Annotations
MetaData	component	Component	0..1	MetaData	component	Component	0..1	[Top Level]	documentDescribes	String - SPDX ID	1..*	If this is not present, all top level compenents are added
MetaData	manufacture	OrganizationalEntity	0..1	MetaData	manufacture	OrganizationalEntity	0..1	[Top Level]	annotations	Annotation Object	0..*	Annotations are used to capture any fields which can not be mapped to SPDX. The Annotation type is OTHER and the comment is of the format MISSING_CDX_PROPERTY: <propertyname>=<propertyJSONvalue>
MetaData	supplier	OrganizationalEntity	0..1	MetaData	supplier	OrganizationalEntity	0..1	CreationInfo	creators	String	1..*	Contact phone numbers and multiple emails care captured in Annotations
MetaData	licenses	LicenseChoice	0..*	MetaData	licenses	LicenseChoice	0..*	[Top Level]	dataLicense	String "CC0"	1..1	Defaults to CC0 if no metadata license is provided
MetaData	properties	Property	0..*	MetaData	properties	Property	0..*	[Top Level]	annotations	Annotation Object	0..*	Annotations are used to capture any fields which can not be mapped to SPDX. The Annotation type is OTHER and the comment is of the format MISSING_CDX_PROPERTY: <propertyname>=<propertyJSONvalue>
								CreationInfo	comment	String	0..1	Not used
Component	type	enum	1..1	Component	type	enum	1..1	Package or File	annotations	Annotation Object	0..*	Annotations are used to capture any fields which can not be mapped to SPDX. The Annotation type is OTHER and the comment is of the format MISSING_CDX_PROPERTY: <propertyname>=<propertyJSONvalue>
Component	mime-type	String	0..1	Component	mime-type	String	0..1	File	fileTypes	enum	0..*	If a mime type does not map to the enum, it is added as an Annotation to the file
Component	bom-ref	String	0..1	Component	bom-ref	String	0..1	Package or File	Id	String	1..1	Converted to a format complying with the SPDX ID syntax
Component	supplier	OrganizationalEntity	0..1	Component	supplier	OrganizationalEntity	0..1	Package	supplier	String	0..1	Contact phone numbers and multiple emails care captured in Annotations
Component	author	String	0..1	Component	author	String	0..1	Package	originator	String	0..1	
Component	publisher	String	0..1	Component	publisher	String	0..1	Package	annotations	Annotation Object	0..*	Annotations are used to capture any fields which can not be mapped to SPDX. The Annotation type is OTHER and the comment is of the format MISSING_CDX_PROPERTY: <propertyname>=<propertyJSONvalue>
Component	group	String	0..1	Component	group	String	0..1	Package	annotations	Annotation Object	0..*	Annotations are used to capture any fields which can not be mapped to SPDX. The Annotation type is OTHER and the comment is of the format MISSING_CDX_PROPERTY: <propertyname>=<propertyJSONvalue>
Component	name	String	1..1	Component	name	String	1..1	Package or File	name	String	1..1	
Component	version	String	1..1	Component	version	String	0..1	Package	versionInfo	String	1..1	
Component	description	String	0..1	Component	description	String	0..1	Package	description	String	1..1	
Component	scope	enum	0..1	Component	scope	enum	0..1	Relationship	relationshipType	enum	1..1	The scope declared on a component is used if the component is included in some type of relationship. Otherwise it is ignored since scope must be taken into account in how the component is used and is not intrinsic to the component itself.
Component	hashes	Hash	0..*	Component	hashes	Hash	0..*	Package or File	checksums	Checksum	1..*	If a checksum algorithm is not supported by SPDX, it is added as an annotation
Component	licenses	LicenseChoice	0..1	Component	licenses	LicenseChoice	0..1	Package or File	declaredLicense or licenseI	AnyLicenseInfo	1..1	Currently, license text information is added as annotations - this can be changed to using ExtractedLicenseInfos in the future
Component	copyright	String	0..1	Component	copyright	String	0..1	Package	copyright	String	1..1	
Component	cpe	String	0..1	Component	cpe	String	0..1	Package	externalRef	ExternalRef	0..*	
Component	purl	String	0..1	Component	purl	String	0..1	Package	externalRef	ExternalRef	0..*	
Component	swid	Swid	0..1	Component	swid	Swid	0..1	Package	annotations	Annotation Object	0..*	Although there is an externalRef which can be used, the Swid object contains the content, so the Annotation approach was chosen
Component	modified	boolean	0..1	Component	modified - deprecated			Package	sourceInfo	String	0..*	Converted to String format
Component	pedigree	Pedigree	0..1	Component	pedigree	Pedigree	0..1	Package	relationships	Relationship	0..*	
Component	externalReferences	ExternalReference	0..*	Component	externalReferences	ExternalReference	0..*	Package	externalRef	ExternalRef	0..*	
Component	components	Component	0..*	Component	components	Component	0..*	Package	relationships	Relationship	0..*	CONTAINS relationship is used for any sub-components
Component	evidence	Evidence	0..1	Component	evidence	Evidence	0..1	Package	attributionTexts	String	0..*	Content is not copied into the attribution texts - this could be done, but it would require more work to understand all supported enodings used by CycloneDX
Component	properties	Property	0..*	Component	properties	Property	0..*	Package or File	annotations	Annotation Object	0..*	Annotations are used to capture any fields which can not be mapped to SPDX. The Annotation type is OTHER and the comment is of the format MISSING_CDX_PROPERTY: <propertyname>=<propertyJSONvalue>
				Component	releaseNotes	ReleaseNote	0..1					New 1.4 fields have not yet been added or analyzed
				Component	signature	JSON Signature	0..1					New 1.4 fields have not yet been added or analyzed
								Package		String	0..*	Not used
								Package	homePage	URL	0..1	Not used
								Package	packageVerificationCode	VerificationCode	0..1	Not used
								Package	downloadLocation	URL	1..1	NOASSERTION value used
								Package	filesAnalyzed	boolean	0..1	Set to false
								Package	licenseComments	String	0..1	Not used
								Package	hasFiles	String	0..*	Not used - Note - if the subcomponents are files, we could add logic to treat those as hasFiles
								Package	comment	String	0..1	Not used
								Package	summary	String	0..1	Not used
								Package	packageFileName	String	0..1	If the component type is a file, then the component name is also added to the packageFileName
								Package	licenseInfosFromFile	AnyLicenseInfo	0..*	Not used
								Package	concludedLicense	AnyLicenseInfo	1..1	Use NOASSERTION
								File	noticeText	String	0..1	Not used
								File	artifactOfs	DoapPorject	0..*	Deprecated - not used
								File	licenseComments	String	0..1	Not used
								File	comment	String	0..1	Not used
								File	fileContributors	String	0..*	Not used
								File	licenseConcluded	AnyLicenseInfo	0..1	NOASSERTION value used
								File	fileDependencies	String	0..*	Deprecated - not used
ExternalReference	url	String	1..1	ExternalReference	url	String	1..1	ExternalRef	referenceLocator	String	1..1	
ExternalReference	comment	String	0..1	ExternalReference	comment	String	0..1	ExternalRef	comment	String	0..1	
ExternalReference	type	enum	1..1	ExternalReference	type	enum	1..1	ExternalRef	referenceType	URI	1..1	most types are mapped to an existing SPDX pre-defined reference type, otherwise a custom reference type is defined with a prefix http://cyclonedx.org/referenctype/
				ExternalReference	hashes	Hash	0..*					New 1.4 fields have not yet been added or analyzed
								ExternalRef	referenceCategory	enum	1..1	Mapped from type