Developer Guide for IBM ConcertDef v1.0.2

Resources

- IBM Concert Documentation
  - https://www.ibm.com/docs/en/concert

- Sample ConcertDef files
  - https://github.com/IBM/Concert/tree/main/use-case-data-files/concertdef-samples/2.0.0.1

- IBM Concert Toolkit
  - https://www.ibm.com/docs/en/concert?topic=methods-using-concert-toolkit

- Sample YAML template files supported by the *Concert Toolkit*
  - https://github.com/IBM/Concert/tree/main/toolkit-enablement/concert-sample/templates

- *Developer Guide* for the *ConcertDef Schema*
  - https://github.com/IBM/Concert/tree/main/toolkit-enablement/concert-utils/concertdef_schema/2.0.0.1

- The command below shows the contents of the Application Inventory SBOM schema in use by the instance at "myhost.com:1234" supporting API key type "C_API_KEY". The API path is "**/core/api/v1/concertdef/schema**" and query parameter for **sbom_type** is **app** <key> and <id> are the API key and instance id in use, respectively. The output can be saved into a local file via the command option "-o".

```
curl -k 'https://myhost.com:1234/core/api/v1/concertdef/schema?sbom_type=app'\
    --header 'Authorization: C_API_KEY <key>' --header 'InstanceId: <id>'
```

- The command below shows how schema validation for a ConcertDef SBOM can be performed via the Concert service in use.

```
curl --location 'https://myhost.com:1234/core/api/v1/validate_concertdef_sbom' \
    --header 'Accept: application/json' \
    --header 'Authorization: C_API_KEY <key>' \
    --header 'InstanceId: <id>' \
    --header 'Content-Type: multipart/form-data' \
    --header 'Content-Type: multipart/form-data' \
    --form 'filename=@"/sample/concert-sample-devsecops-app-components.json"'
```

Documentation Modification Date: 2025-08-19

Contents

1 Overview

IBM Concert promises automatically generating "*Application 360 Insights*" when sufficient data is loaded into it. Sample factual data regard software development lifecycle tasks, *software bill-of-materials* (***SBOMs***), deployment evidences, endpoint certificates, continuous vulnerability scan results, and continuous compliance posture measures.

The ConcertDef JSON schema represents the Concert-defined data model to generate a 360-degree topological view of your applications and environments' schema contains a subset of application component-type extensions, including a *properties* object with details about the change to the CI/CD pipeline.

Application Sboms

- An Application SBOM is always treated as a full definition of your application
- If you want to add a component to your application, then the application SBOM should include existing component definition as well the new component definition
- If you upload the same application SBOM with new set of components, previous components will get removed.
- Application SBOM now supports build & deploy sboms as well and there is no need to have build & deploy SBOM separately anymore.

There are two variants of Application sboms

1. ***Application SBOM with Application components***

   - ***Components*** section will have the information about ***Container Images***/***Libraries*** and ***Code Repos***
   - One ***Build Section*** can include one or more ***Code Repos***, which can be used to build one or more ***Container Images***.
   - One ***Build Section*** can include one ***Container Image*** built via several ***Code Repos***.
   - If one ***Build Section*** has one ***Container Image/Library*** and have multiple ***Code Repos*** they can be associated automatically
   - If one ***Build Section*** have Multiple ***Container Images*** and ***Code Repos,*** they can be linked through ***Dependencies*** section.

2. ***Application SBOM with Runtime components***

   - ***Runtime-Components*** section will have the information about ***Deployed Container Images***/***Libraries***
   - ***Deployment section*** can have one or more ***Runtime-Components***
   - Several ***Container Images*** can be deployed to one ***Deployment Environment***
   - ***One Container Image*** can be deployed to several ***Deployment Environments***

An Application SBOM contains a set of structured Inventory SBOM selectors that enable business-oriented, 360-degree views of selected Build and/or Deploy Inventories. These views integrate other evidence data (e.g., package SBOMs, vulnerability scan results, endpoint certificates) along with business support data (e.g., business organization details, application criticality settings).

These typed *ConcertDef* SBOM files must be formatted in JSON when they are loaded into a specific instance of Concert via the instance's Console GUI or the instance's API service endpoint. The typed JSON files are usually named as "app" SBOMs

A complete *ConcertDef JSON Schema* specification is available at the URL below.
https://github.com/IBM/Concert/tree/main/toolkit-enablement/concert-utils/concertdef_schema/2.0.0.1
It can be used to check for *syntax* errors of the JSON-formatted ConcertDef SBOMs (via tools like `check-jsonschema`) and to enable schema-assisted editing of the ConcertDef files via modern text editors like `vscode`. Home page of the tool `check-jsonschema` is at https://github.com/python-jsonschema/check-jsonschema
A good how-to summary for schema-based editing in `vscode` is available at:
https://frontaid.io/blog/json-schema-vscode/

One JSON Schema specification (with `description` fields and `examples` objects) is provided for each type of ConcertDef SBOM files. They are more consumable than the complete ConcertDef Schema specification and can detect more type-specific schema errors.

This document provides two sample ConcertDef SBOMs for a sample application named "`concert-sample-devsecops`" in the appendix. Required JSON objects within a specific JSON object are highlighted using colors, e.g., `.bomFormat` is colored because that is required for all the ConcertDef SBOMs.

- Appendix A: Sample Application with application components SBOM
- Appendix B: Sample Application with runtime-components SBOM

Every `component` or `service` JSON object in a ConcertDef SBOM must have a `name`.
Every ConcertDef SBOM must have three top-level JSON objects:

- `bomFormat`: It must be a string value "`ConcertDef`".
- *`specVersion`*: It must be a string value "`1.0.2`".
- *`metadata`*: An object of type "`application`".

Other sample ConcertDef SBOMs are available at:
https://github.com/IBM/Concert/tree/main/use-case-data-files/concertdef-samples/2.0.0.1

2 ConcertDef JSON Schema

ConcertDef Schema is a *JSON Schema* specification for *ConcertDef SBOMs*. Custom extensions of the JSON-based "syntax" specification can be done via JSON Schema `properties` objects, which are *JSON* `arrays` of two-field objects, each of which comprises `name` and `value` fields. After a specific *ConcertDef SBOM* is loaded into the Concert service in use, "semantic" errors are checked by the service. Additionally, the Concert service links the data included in the loaded ConcertDef files with the data in other types of files that Concert supports, such as CycloneDX package SBOMs, CVE vulnerability scans, endpoint certificates, and compliance postures. Finally, the service links all the data with relevant built-in and 3rd-party data, e.g., National Vulnerability Database (NVD) data.

Type-specific ConcertDef Schema specifications can be accessed from the Concert service instance in use via the CURL command. For example, the command below shows the contents of the Application Inventory SBOM schema in use by the instance at "myhost.com:1234" supporting API key type "C_API_KEY". The API path is "**/core/api/v1/concertdef/schema**" and query parameter for **sbom_type** is **app**. <key> and <id> are the API key and instance id in use, respectively. The output can be saved into a local file via the command option "-o".

```
curl -k 'https://myhost.com:1234/core/api/v1/concertdef/schema?sbom_type=app'
--header 'Authorization: C_API_KEY <key>' --header 'InstanceId: <id>'
```

The CURL command below exemplifies how schema validation for a ConcertDef SBOM can be performed via the Concert service in use.

```
curl --location 'https://myhost.com:1234/core/api/v1/validate_concertdef_sbom' \
    --header 'Accept: application/json' \
    --header 'Authorization: C_API_KEY <key>' \
    --header 'InstanceId: <id>' \
    --header 'Content-Type: multipart/form-data' \
    --form 'filename=@"/sample/concert-sample-devsecops-app.json"'
```

JSON-formatted *ConcertDef SBOMs* can be generated via ConcertDef YAML templates. The *Concert Toolkit image* can be used to transform a YAML-formatted ConcertDef template file into a *ConcertDef SBOM*. More details about the IBM Concert Toolkit are available at: https://www.ibm.com/docs/en/concert?topic=methods-using-concert-toolkit

Sample YAML template files supported by the *Concert Toolkit* are available at: https://github.com/IBM/Concert/tree/main/toolkit-enablement/concert-sample/templates

2.1 Application SBOMs with Application components

The top-level JSON objects that are specific to Application SBOMs are listed below.

- `components`: An array of typed *build* definition. The `name`, `version` are used by Concert to compose an internal unique id for the Build definition.
- `components->components`: An array of typed *build* objects listed below. It must include at least one `container` or `library or source code` object. This is the place where you can co-relate the build objects with application.
  - o `code`: A structured set of inventory data for a **source code repository**.
  - o `container`: A structured set of inventory data for a **container image**.
  - o `library`: A structured set of inventory data for a **software library**.
- `properties`: An array of application-level properties, e.g., application criticality
- `tags`: An optional array of strings that can be used for searching or filtering.
- Environments: An array of deployed environments and its access points details
- Services: A list of services
- Dependencies: Defines the direct dependencies of a component or service

Appendix A shows a sample Application SBOM with application components: `metadata` and `components`.

- `metadata`: The `type` field (on line 6) is required, and `"application"` must be the value. The `name` (on line 8) and `version` (on line 9) are set for the ConcertDef application. Contents of the `business` object are determined by the business manager of the Concert service account in use.
- `components`: An array of the `build` *objects* selected in terms of the `name` and `version` for unique identification.
- `Components->`*`components`*: This sample exemplifies the use of two container-related objects:
  - o `container`: Its `name` (on line 31) is the image path (without tag or digest), which includes the image repository name as the last naming component delimited by character "/". Its `uri` (on line 33) is *unique resource identifier* composed of the *image path*, *image tag*, and/or *image digest*. The format is "<path>:<tag>", "<path>@<digest>" or "<path>:<tag>@<digest>". That is, *image path* (a.k.a., image repository path) is required. An *image tag* labels a specific image instance in the image repository mainly for human consumption, and an *image digest* is a *content-based* version indicator for a specific image instance. When this `uri` field is not included in the SBOM, Concert composes the identifier in terms of the *defined* values of `name`, `tag`, and `digest`. In practice, `latest` is the default tag value when neither tag nor digest is specified, though common best practices do not use that to tag image instances.
  - o `code`: Its `name` (on line 54) is associated with its `purl` (on line 55), which refers to the source code repository/filesystem used for creating the `container` object. When a git repository is used for hosting the source code, `commit_sha` (on line 56) is needed to uniquely identify the version of the source code (as shown in this sample). Without `commit_sha`, Concert cannot know which version of the repository was used for the generation of a specific source-code-based CVE vulnerability scan. For non-git use cases, any version string can be used as the value for

`commit_sha`. `branch` (on line 57) is optional. That is used to facilitate human consumption, and cannot be used as a unique version identifier.

`cyclonedx-bom-link` (on line 58) is the unique *BOM-Link* of a CycloneDX SBOM. The sample exemplifies how a source-code-based generation of a CycloneDX *Package SBOM* can be associated with a Source code repository. `cyclonedx-bom-link` would be included in the `container` object if an image-based CycloneDX *Package SBOM* was generated for the container image.

2.2 Application SBOMs with Runtime-components

Below shows sample Application sbom with Runtime-components.

- `runtime-components`: An array of inventory objects for deployment runtime
  - `kubernetes`: A Kubernetes-based container runtime.
  - `vm`: A VM or server deployment runtime.
  - `zOS`: A zOS deployment runtime.
- `services`: An array of inventory objects for API services
- `dependencies`: An array of dependency relationship objects
  - `ref`: A `bom-ref` which uniquely identifies a `component` or `service` within the SBOM.
  - `dependsOn`: A `bom-ref` array of the dependent components and/or services
- `properties`: An array of name-value objects
- `tags`: An array of strings that can be used for searching or filtering.

The JSON objects listed in Appendix B are detailed below.

- `metadata`: The `type` field (on line 6) is required, and `"application"` must be the value. The `name` (on line 8) and `version` (on line 9) are set for the ConcertDef application. Contents of the `business` object are determined by the business manager of the Concert service account in use.
- `Runtime-components`: This will have the cluster details and deployed images information.

The fields `type` (one line 25), `name` (on line 26) are required for the `runtime-components` object on line 22. Value of the type is `"kubernetes"`. Value of the `name` in this sample is the value of `cluster_id` (on line 38), which is unique in the `cluster_platform` `"roks"` (on lines 34).

The `namespaces` object (on line 50) is an array of cluster namespaces and labels that consist of the application.

The `components` object (on line 64) is an array of `namespace` objects, each of which includes an array of `container` or `library` objects as its `components`. For example, line 67 shows namespace `"concert-instance-1"` is used for deploying a container image with its URI specified at line 73.

The `container` objects deployed to several deployment runtimes. Moreover, every deployment runtime can include several `namespace` objects, each of which can be used to deploy several container images.

The `services` object (on line 106) exemplifies how the base URL (on line 90) and network exposure (on line 115) of an API service (named on line 109) can be included. The `dependencies` object (on line 121) relates the `container` object (referenced on line 123) with the API service object (referenced on line 125) via their `bom-ref` values.

3 Linking ConcertDef SBOM Data with Other Concert Data

ConcertDef SBOMs enable integration of various siloed enterprise data, such as vulnerability scan results, package SBOMs (formatted in CycloneDX), endpoint certificates, and runtime performance and compliance postures:

- **Code based vulnerability CVEs** are linked with *code* objects
- **Image based vulnerability CVEs** are linked with the *container* objects
- **Package SBOMs** sourced from source (or images) are linked with the *code* (or *con-tainer*) objects
- **Endpoint certificates** are linked with the HTTPS endpoints (or base URLs)
- **Runtime performance and compliance postures** for Kubernetes-like clusters are linked with the *kubernetes* objects

We note that when a specific code-based package SBOM is loaded, the associated code repository URL (or pathname of the source tree root) must be provided as metadata such that the uploaded SBOM can be listed in the Console's *software composition* GUI.

A. Sample Application SBOM with Application components

```json
{
  "bomFormat": "ConcertDef",
  "specVersion": "1.0.2",
  "metadata": {
    "timestamp": "2024-12-06T15:23:57Z",
    "type": "application",
    "component": {
      "name": "concert-sample-devsecops",
      "version": "1.0.0"
    },
    "business": {
      "name": "Acme Inc.",
      "units": [
        {
          "name": "Unit 1",
          "email": "myemail@acme.com",
          "phone": "(123) 123-1234"
        }
      ]
    }
  },
  "components": [
    {
      "bom-ref": "build:concert-sample-devsecops",
      "type": "build",
      "name": "concert-sample-devsecops",
      "version": "1.0.0",
      "components": [
        {
          "bom-ref": "container:us.icr.io/icr4mcc/concert-sample-devsecops",
          "type": "container",
          "name": "us.icr.io/icr4mcc/concert-sample-devsecops",
          "uri": "us.icr.io/icr4mcc/concert-sample-devsecops@sha256:62b2e97c78d6142c12d94f8077efc5c1f40888c3e473e1e2e3a7a8700c3b9a54",
          "tag": "2.0.0",
          "digest": "sha256:62b2e97c78d6142c12d94f8077efc5c1f40888c3e473e1e2e3a7a8700c3b9a54",
          "properties": [
            {
              "name": "base_image",
              "value": "ubi9"
            }
          ]
        },
        {
          "bom-ref": "container:us.icr.io/icr4mcc/concert-test-devsecops",
          "type": "container",
          "name": "us.icr.io/icr4mcc/concert-test-devsecops",
          "uri": "us.icr.io/icr4mcc/concert-test-devsecops@sha256:22b2e97c78d6142c12d94f8077efc5c1f40888c3e473e1e2e3a7a8700c3b9a54",
          "tag": "2.0.0",
          "digest": "sha256:22b2e97c78d6142c12d94f8077efc5c1f40888c3e473e1e2e3a7a8700c3b9a54"
        },
        {
          "bom-ref": "repository:coderepo:github:concert-sample-devsecops",
          "type": "code",
          "name": "concert-sample-devsecops",
          "purl": "https://github.ibm.com/concert/concert-sample-devsecops",
          "commit_sha": "1a2b3c4d5e6f7890123456789abcdef012345678",
          "branch": "main",
          "cyclonedx-bom-link": "urn:uuid:eeff39fd-91d0-42f2-836e-c73be49b8112/1",
          "properties": [
            {
              "name": "version_control",
              "value": "github"
            }
          ]
        },
        {
          "bom-ref": "repository:coderepo:github:concert-test-devsecops",
          "type": "code",
          "name": "concert-test-devsecops",
          "purl": "https://github.ibm.com/concert/concert-test-devsecops",
          "commit_sha": "2a2b3c4d5e6f7890123456789abcdef012345678",
          "branch": "main"
        }
      ]
    }
  ],
  "environments": [
    {
      "bom-ref": "environment:dev",
      "type": "environment",
      "name": "dev",
      "access-points": [
        {
          "ref": "appapi:concert-sample-devsecops",
          "base-url": "https://internal.mycompany.io/concert-sample-devsecops"
        }
      ],
      "properties": [
        {
          "name": "platform_type",
          "value": "ec2"
        },
        {
          "name": "deployment_purpose",
          "value": "development"
        }
      ]
    },
    {
      "bom-ref": "environment:stage",
      "type": "environment",
      "name": "stage"
    },
    {
      "bom-ref": "environment:prod",
      "type": "environment",
      "name": "prod"
    }
  ],
  "services": [
    {
      "bom-ref": "appapi:concert-sample-devsecops",
      "name": "concert-sample-devsecops",
      "endpoints": [
        "/"
      ],
      "properties": [
        {
          "name": "network_exposure",
          "value": "public"
        }
      ]
    }
  ],
  "dependencies": [
    {
      "ref": "container:us.icr.io/icr4mcc/concert-sample-devsecops",
      "dependsOn": [
        "appapi:concert-sample-devsecops",
        "repository:coderepo:github:concert-sample-devsecops"
      ]
    },
    {
      "ref": "container:us.icr.io/icr4mcc/concert-test-devsecops",
      "dependsOn": [
        "repository:coderepo:github:concert-test-devsecops"
      ]
    }
  ],
  "properties": [
    {
      "name": "application_criticality",
      "value": "4"
    },
    {
      "name": "application_data_assessment_impact_risk",
      "value": "5"
    },
    {
      "name": "revenue",
      "value": "1000"
    }
  ],
  "tags": [
    "sample"
  ]
}
```

B. Sample Application SBOM with Runtime components

```
{
  "bomFormat": "ConcertDef",
  "specVersion": "1.0.2",
  "metadata": {
    "timestamp": "2024-12-06T15:23:57Z",
    "type": "application",
    "component": {
      "name": "concert-sample-devsecops",
      "version": "1.0.0"
    },
    "business": {
      "name": "Acme Inc.",
      "units": [
        {
          "name": "Unit 1",
          "email": "myemail@acme.com",
          "phone": "(123) 123-1234"
        }
      ]
    }
  },
  "runtime-components": [
    {
      "bom-ref": "service:kubernetes:rosa:development-k8-1",
      "name": "development-k8-1",
      "type": "kubernetes",
      "api-server": "https://api.rosa1.1jk4.p1.openshiftapps.com:6443",
      "properties": [
        {
          "name": "platform",
          "value": "rosa"
        },
        {
          "name": "cluster_platform",
          "value": "aws"
        },
        {
          "name": "cluster_region",
          "value": "us-east-1"
        },
        {
          "name": "cluster_name",
          "value": "development-k8-1"
        }
      ],
      "namespaces": [
        {
          "namespace": "concert-instance-1",
          "labels": [
            "label1"
          ]
        },
        {
          "namespace": "concert-instance-2",
          "labels": [
            "label2"
          ]
        }
      ],
      "components": [
        {
          "type": "namespace",
          "name": "concert-instance-1",
          "components": [
            {
              "bom-ref": "container:us.icr.io/sample/concert-sample",
              "type": "container",
              "name": "us.icr.io/icr4mcc/concert-sample-devsecops",
              "uri": "us.icr.io/icr4mcc/concert-sample-devsecops@sha256:62b2e97c78d6142c12d94f8077efc5c1f40888c3e473e1e2e3a7a8700c3b9a54",
              "tag": "2.0.0",
              "digest": "sha256:62b2e97c78d6142c12d94f8077efc5c1f40888c3e473e1e2e3a7a8700c3b9a54"
            }
          ]
        }
      ]
    }
  ],
  "environments": [
    {
      "bom-ref": "environment:dev",
      "type": "environment",
      "name": "dev",
      "access-points": [
        {
          "ref": "appapi:concert-sample-devsecops",
          "base-url": "https://internal.mycompany.io/concert-sample-devsecops",
          "port": 8000
        }
      ],
      "properties": [
        {
          "name": "platform_type",
          "value": "ec2"
        },
        {
          "name": "deployment_purpose",
          "value": "development"
        }
      ]
    }
  ],
  "services": [
    {
      "bom-ref": "appapi:concert-sample-devsecops",
      "name": "concert-sample-devsecops",
      "endpoints": [
        "/"
      ],
      "properties": [
        {
          "name": "network_exposure",
          "value": "public"
        }
      ]
    }
  ],
  "dependencies": [
    {
      "ref": "container:us.icr.io/sample/concert-sample",
      "dependsOn": [
        "appapi:concert-sample-devsecops"
      ]
    }
  ],
  "properties": [
    {
      "name": "application_criticality",
      "value": "4"
    },
    {
      "name": "application_data_assessment_impact_risk",
      "value": "5"
    },
    {
      "name": "revenue",
      "value": "1000"
    }
  ],
  "tags": [
    "sample"
  ]
}
```