Developer Guide for IBM ConcertDef v1.0.2

1. Overview

IBM Concert promises automatically generating "*Application 360 Insights*" when sufficient data is uploaded to it. Its core data model comprises three types of Concert-defined (ConcertDef) data in terms of:

1. *Build Inventories*
   - E.g., evidence data for a specific *Container Image* and the *Code Repo* used to build it
2. *Deploy Inventories*
   - E.g., evidence data for a specific Kubernetes cluster and the *Container Images* deployed to it
3. *Application Blueprints*
   - E.g., selection criteria for in-scope *Build Inventories* and *Deployment Environments*

For container-based use cases, the following is a summary of the relationships between *Build Inventories*, *Deploy Inventories,* and *Application Blueprints.*

- One ***Build Inventory*** can include only one ***Code Repo***, which can be used to build one or more ***Container Images***.
- One ***Build Inventory*** can include only one ***Container Image*** built via several ***Code Repos***.
- Every ***Deploy Inventory*** is bound with one and only one ***Deployment Environment***, which may comprise several Kubernetes clusters and namespaces.
- Several ***Container Images*** can be deployed to one ***Deployment Environment*** evidenced by one ***Deploy Inventory***.
- One ***Container Image*** can be deployed to several ***Deployment Environments*** evidenced via several ***Deploy Inventories***.
- *One **Application Blueprint*** can select (without details) several in-scope ***Deployment Environments***.
- *One* **Build Inventory** can be associated with several ***Application Blueprints*** per the name and version specified in its metadata.
- *One **Application Blueprint*** can select (without details) several ***Build Inventories*** in terms of their (component) names and the versions.

These typed *ConcertDef* data must be formatted in JSON when they are uploaded to 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 "build", "deploy", and "application" SBOMs, respectively. The *JSON Schema* specification for the ConcertDef files (with *description* fields and *examples* objects) is available at: https://github.ibm.com/roja/concert-utils/tree/main/concertdef_schema

The *ConcertDef JSON Schema* can be used to check for *syntax* errors of the JSON-formatted ConcertDef files (via tools like `check-jsonschema`) and to enable schema-assisted editing of the ConcertDef files via modern text editors like `vscode`. To facilitate the use of the *ConcertDef Schema*, three sub-schema specifications are available for each type of ConcertDef files. Sample ConcertDef files are available at: https://www.ibm.com/docs/en/concert?topic=inventory-generating-concert-defined-sbom

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*. The home page URL of the IBM Concert Toolkit is: https://github.ibm.com/roja/toolkit

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 uploaded to the Concert service in use, "semantic" errors are checked by the service, e.g., unsupported *specVersion* string for the *ConcertDef Schema* in use. The service also links the data included in the uploaded ConcertDef files and in other types of Concert files. Finally, the service links all the data with relevant built-in and 3rd-party data, e.g., National Vulnerability Database (NVD) data.

With reference to the *ConcertDef Schema* specification, every *component* or *service* JSON object must have a *name*. Every ConcertDef SBOM must have three top-level JSON objects:

- *bomFormat*: Its string value must be "ConcertDef".
- *specVersion*: As of to date, the specification version string must be "1.0.2".
- *metadata*: A typed object with the value of its *type* as "build", "deploy", or "application".

3. ConcertDef Build Inventory SBOMs

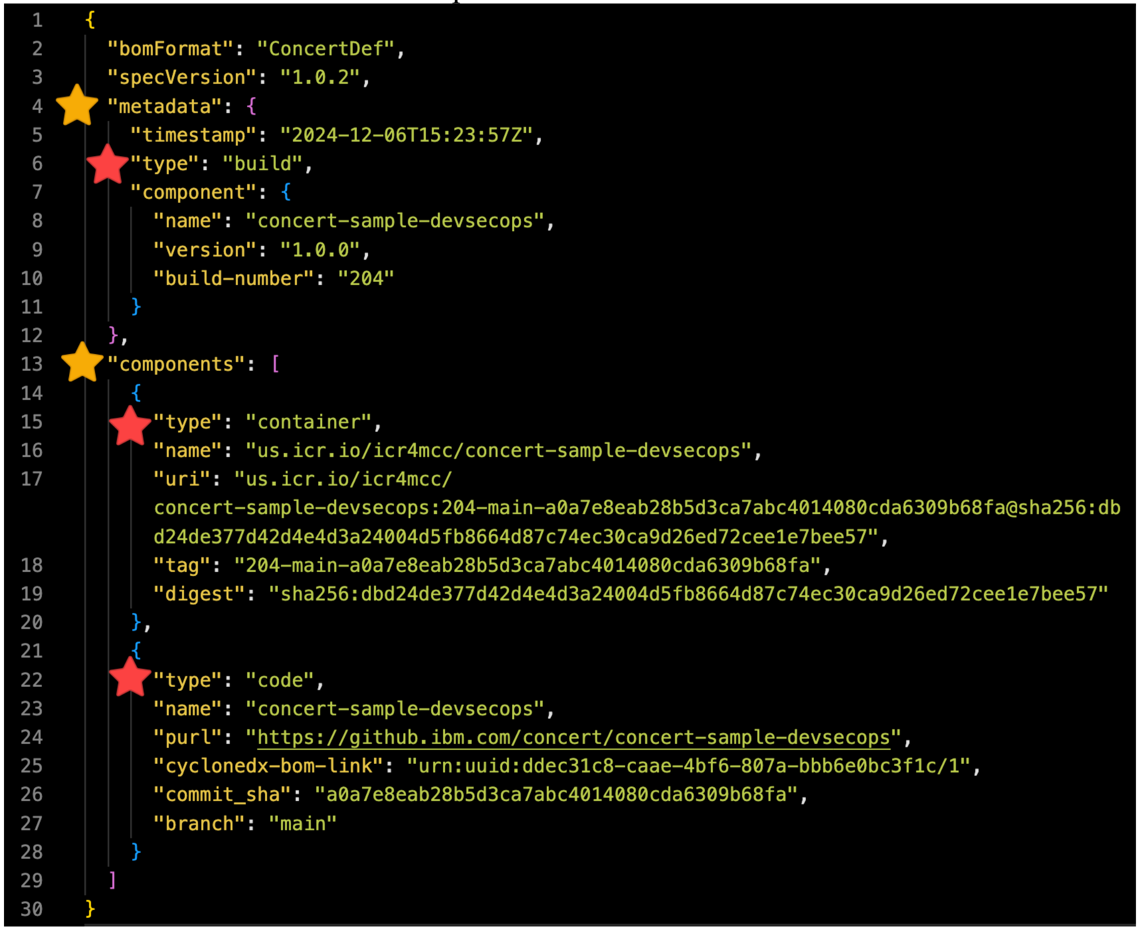
The top-level JSON objects that are specific to ConcertDef *Build Inventory SBOMs* are listed below. Per the *ConcertDef Schema* specification, all of them are syntactically optional.

- *components*: An array of typed *build inventory* objects listed below.
  - *container*: A structured set of inventory data for a **container image**.
  - *code*: A structured set of inventory data for a **source code repository**.
  - *library*: A structured set of inventory data for a **software library**.
- *properties*: An array of name-value objects.
- *tags*: An optional array of strings that can be used for searching or filtering.

Figure 1 shows a sample ConcertDef *Build Inventory SBOM*. Build-specific top-level objects are:

- *metadata*: The *type* field (on line 6) is required, and the value must be "build". The SBOM creation *timestamp* (on line 5) and *build-number* (on line 10) are optional. The *name* (on line 8) and *version* (on line 9) are used by a ConcertDef *Application SBOM* to select this *Build SBOM*.
- *components*: This sample exemplifies the use of two container-related objects:
  - *container*: Its *name* (on line 16) is the image registry path. Its *uri* (on line 17) is a unique resource identifier composed of the image registry path, image tag, and image digest. When this field is not included in the SBOM, Concert composes the identifier in terms of the values of *name*, *tag*, and *digest*.
  - *code*: Its *name* (on line 23) is associated with its *purl* (on line 24) in Concert, 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_shar* (on line 26) is needed to uniquely identify the version of the source code (as shown in this sample). *branch* (on line 27) is optional, and cannot be used as a unique version identifier.

In Figure 1, *cyclonedx-bom-link* on line 25 is the unique *BOM-Link* of a CycloneDX SBOM. The sample exemplifies how a source-code-based generation of a CycloneDX *Package SBOM* is associated with a ConcertDef *Build SOM*. *cyclonedx-bom-link* would be included in the *container* object if an image-based CycloneDX *Package SBOM* was generated for the container image. In the Concert console, *.metadata.component.name* (on line 8) is the component name associated with the uploaded SBOM.



```
{
  "bomFormat": "ConcertDef",
  "specVersion": "1.0.2",
  "metadata": {
    "timestamp": "2024-12-06T15:23:57Z",
    "type": "build",
    "component": {
      "name": "concert-sample-devsecops",
      "version": "1.0.0",
      "build-number": "204"
    }
  },
  "components": [
    {
      "type": "container",
      "name": "us.icr.io/icr4mcc/concert-sample-devsecops",
      "uri": "us.icr.io/icr4mcc/
      concert-sample-devsecops:204-main-a0a7e8eab28b5d3ca7abc4014080cda6309b68fa@sha256:db
      d24de377d42d4e4d3a24004d5fb8664d87c74ec30ca9d26ed72cee1e7bee57",
      "tag": "204-main-a0a7e8eab28b5d3ca7abc4014080cda6309b68fa",
      "digest": "sha256:dbd24de377d42d4e4d3a24004d5fb8664d87c74ec30ca9d26ed72cee1e7bee57"
    },
    {
      "type": "code",
      "name": "concert-sample-devsecops",
      "purl": "https://github.ibm.com/concert/concert-sample-devsecops",
      "cyclonedx-bom-link": "urn:uuid:ddec31c8-caae-4bf6-807a-bbb6e0bc3f1c/1",
      "commit_sha": "a0a7e8eab28b5d3ca7abc4014080cda6309b68fa",
      "branch": "main"
    }
  ]
}
```

Figure 1. A Sample ConcertDef Build Inventory SBOM.

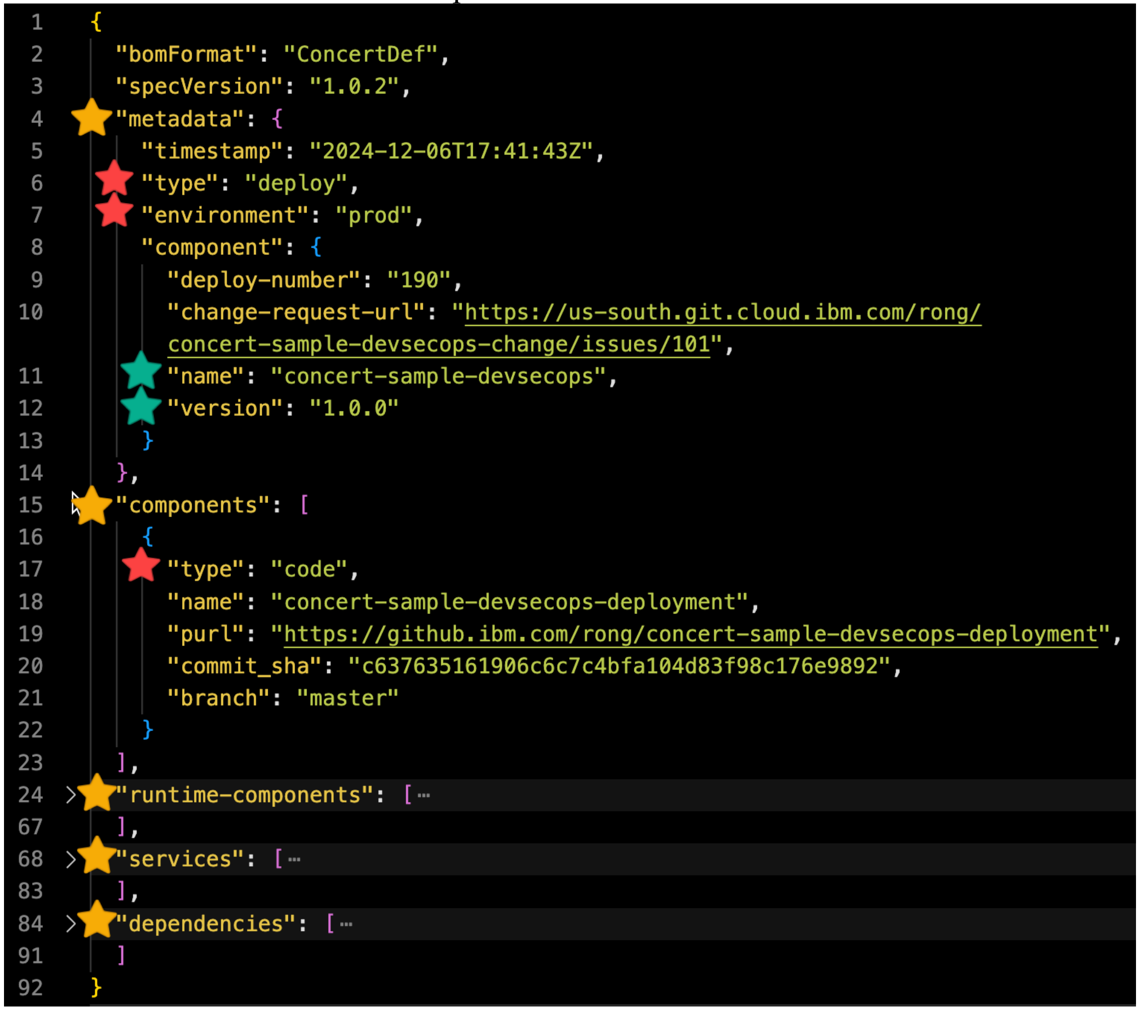
4. ConcertDef Deploy Inventory SBOMs

The top-level JSON objects that are specific to ConcertDef *Deploy SBOMs* are:

- *components*: An array of inventory objects for deployment repositories
  - *code*: A structured set of inventory data for a deployment repository
- *runtime-components*: An array of inventory objects for deployment runtime components
  - *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 identify a component or service within the BOM.
  - *dependsOn*: An array of *bom-ref* identifiers of the dependent components or services
- *properties*: An array of name-value objects
- *tags*: An array of strings that can be used for searching or filtering.

Figure 2 shows an incomplete ConcertDef *Deploy Inventory SBOM*. Deploy-specific top-level objects are:

- *metadata*: The *type* field (on line 6) is required, and the value must be "deploy". The *environment* field (on line 7) must be set, and the value must be one of the deployment environment names determined by the business manager for the Concert service account in use. The SBOM creation



```
 1  {
 2    "bomFormat": "ConcertDef",
 3    "specVersion": "1.0.2",
 4    "metadata": {
 5      "timestamp": "2024-12-06T17:41:43Z",
 6      "type": "deploy",
 7      "environment": "prod",
 8      "component": {
 9        "deploy-number": "190",
10        "change-request-url": "https://us-south.git.cloud.ibm.com/rong/
          concert-sample-devsecops-change/issues/101",
11        "name": "concert-sample-devsecops",
12        "version": "1.0.0"
13      }
14    },
15    "components": [
16      {
17        "type": "code",
18        "name": "concert-sample-devsecops-deployment",
19        "purl": "https://github.ibm.com/rong/concert-sample-devsecops-deployment",
20        "commit_sha": "c637635161906c6c7c4bfa104d83f98c176e9892",
21        "branch": "master"
22      }
23    ],
24  > "runtime-components": [⋯
67    ],
68  > "services": [⋯
83    ],
84  > "dependencies": [⋯
91    ]
92  }
```

Figure 2. An Incomplete ConcertDef Deploy Inventory SBOM.

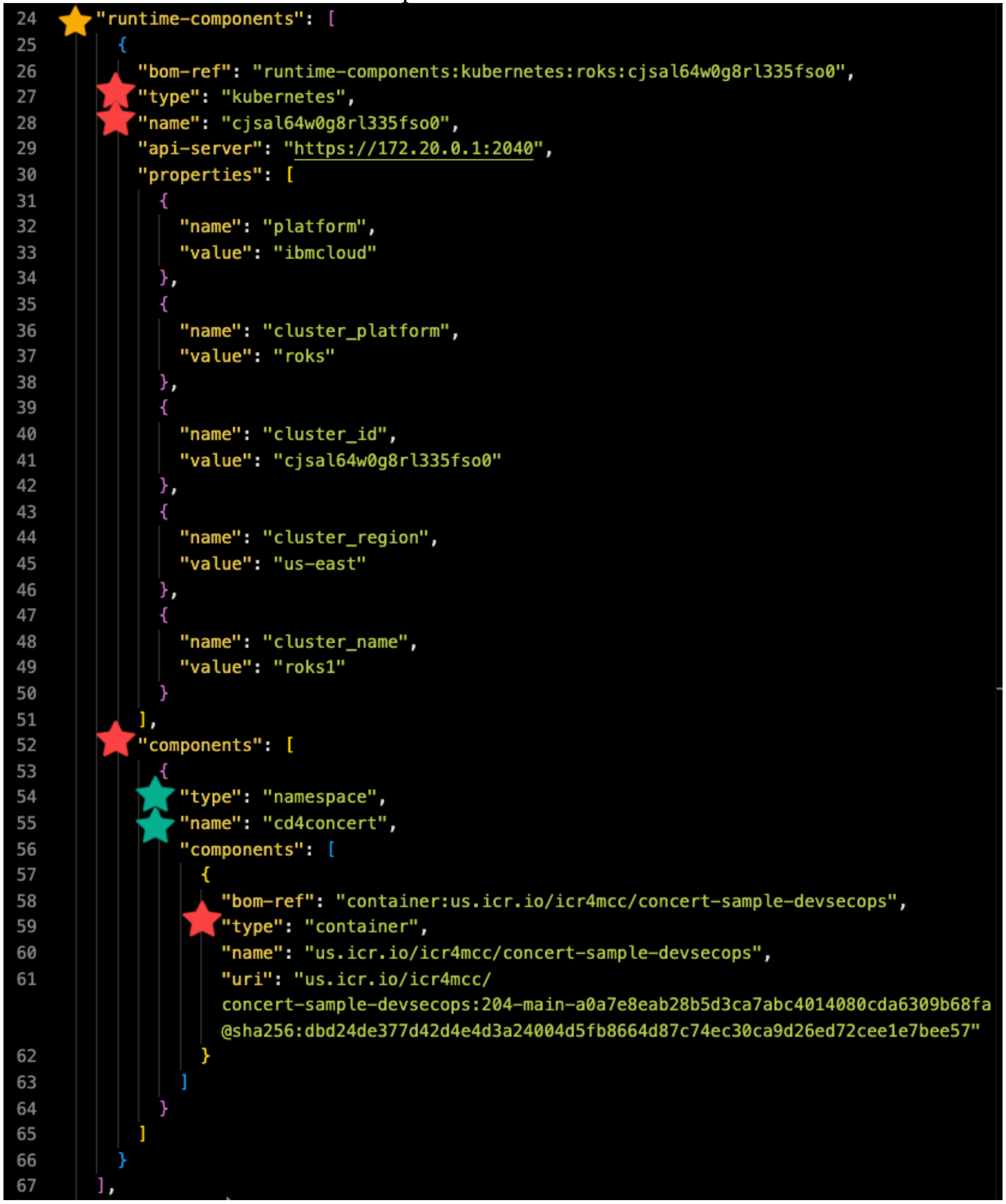
timestamp (on line 5), *deploy-number* (on line 9), and *change-request-url* (on line 10) are optional. The *name* (on line 11) and *version* (on line 12) are used by a ConcertDef *Application SBOM* to select this *Deploy SBOM*.

- *components*: This sample exemplifies the inclusion of a *code* object for the deployment repo (or file system) that hosts the deployment artifacts (e.g., scripts, Kubernetes manifests, Helm charts, etc.).

Figure 3 shows the *runtime-components* object (of type "kubernetes") of the sample ConcertDef *Deploy Inventory SBOM*. The fields *type* (one line 27), *name* (on line 28), and *components* (on line 52) are required. Value of the *name* in this case is the value of *cluster_id* (on line 40), which is unique in the cluster_platform "roks" (on lines 37).

The *components* object (on line 52) is an array of *namespace* objects, each of which includes an array of *container* objects as its *components* (on line 56). For example, line 55 shows namespace "cd4concert" is used for deploying a container image with its URI specified at line 61.

We note that since each *runtime-components* object is bound to a typed deployment runtime, one *Deploy SBOM* can comprise 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. As per the ConcertDef data model, every *Deploy SBOM* is bound with one and only one ConcertDef *environment* defined by the business manager of the Concert service account in use.

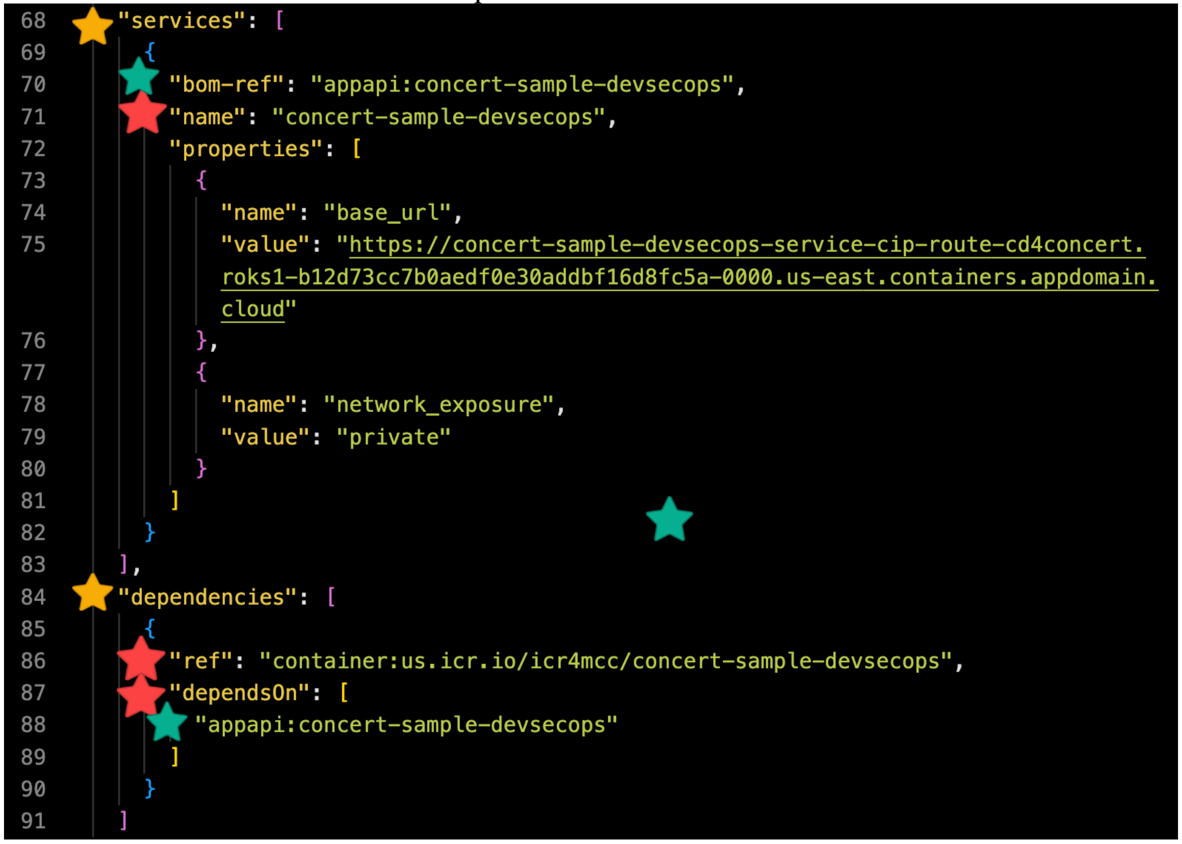


```
"runtime-components": [
  {
    "bom-ref": "runtime-components:kubernetes:roks:cjsal64w0g8rl335fso0",
    "type": "kubernetes",
    "name": "cjsal64w0g8rl335fso0",
    "api-server": "https://172.20.0.1:2040",
    "properties": [
      {
        "name": "platform",
        "value": "ibmcloud"
      },
      {
        "name": "cluster_platform",
        "value": "roks"
      },
      {
        "name": "cluster_id",
        "value": "cjsal64w0g8rl335fso0"
      },
      {
        "name": "cluster_region",
        "value": "us-east"
      },
      {
        "name": "cluster_name",
        "value": "roks1"
      }
    ],
    "components": [
      {
        "type": "namespace",
        "name": "cd4concert",
        "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:204-main-a0a7e8eab28b5d3ca7abc4014080cda6309b68fa
            @sha256:dbd24de377d42d4e4d3a24004d5fb8664d87c74ec30ca9d26ed72cee1e7bee57"
          }
        ]
      }
    ]
  }
],
```

Figure 3. runtime-components object of a Sample ConcertDef Deploy Inventory SBOM.

Figure 4 shows the *services* and *dependencies* objects of the sample *Deploy SBOM*. The *services* object exemplifies how the base URL (on line 75) and network exposure (on line 79) of an API service (named on line 71) can be included in a *Deploy SBOM*. The *dependencies* object (on line 84) relates the *container* object (referenced on line 86) with the API service object (referenced on line 88) via their *bom-ref* values.

We note that network exposure settings (either "public" or "private") can be specified in ConcertDef *Application Blueprint SBOMs*. The default value is "private". When the network exposure setting for a specific API access endpoint in a *Deploy Inventory SBOM* is different from the *Application Blueprint SBOM* that selects the *Deploy Inventory SBOM*, Concert uses the one specified in the *Application SBOM*.



```
"services": [
  {
    "bom-ref": "appapi:concert-sample-devsecops",
    "name": "concert-sample-devsecops",
    "properties": [
      {
        "name": "base_url",
        "value": "https://concert-sample-devsecops-service-cip-route-cd4concert.roks1-b12d73cc7b0aedf0e30addbf16d8fc5a-0000.us-east.containers.appdomain.cloud"
      },
      {
        "name": "network_exposure",
        "value": "private"
      }
    ]
  }
],
"dependencies": [
  {
    "ref": "container:us.icr.io/icr4mcc/concert-sample-devsecops",
    "dependsOn": [
      "appapi:concert-sample-devsecops"
    ]
  }
]
```

Figure 4. services and dependencies objects of a Sample ConcertDef Deploy Inventory SBOM.

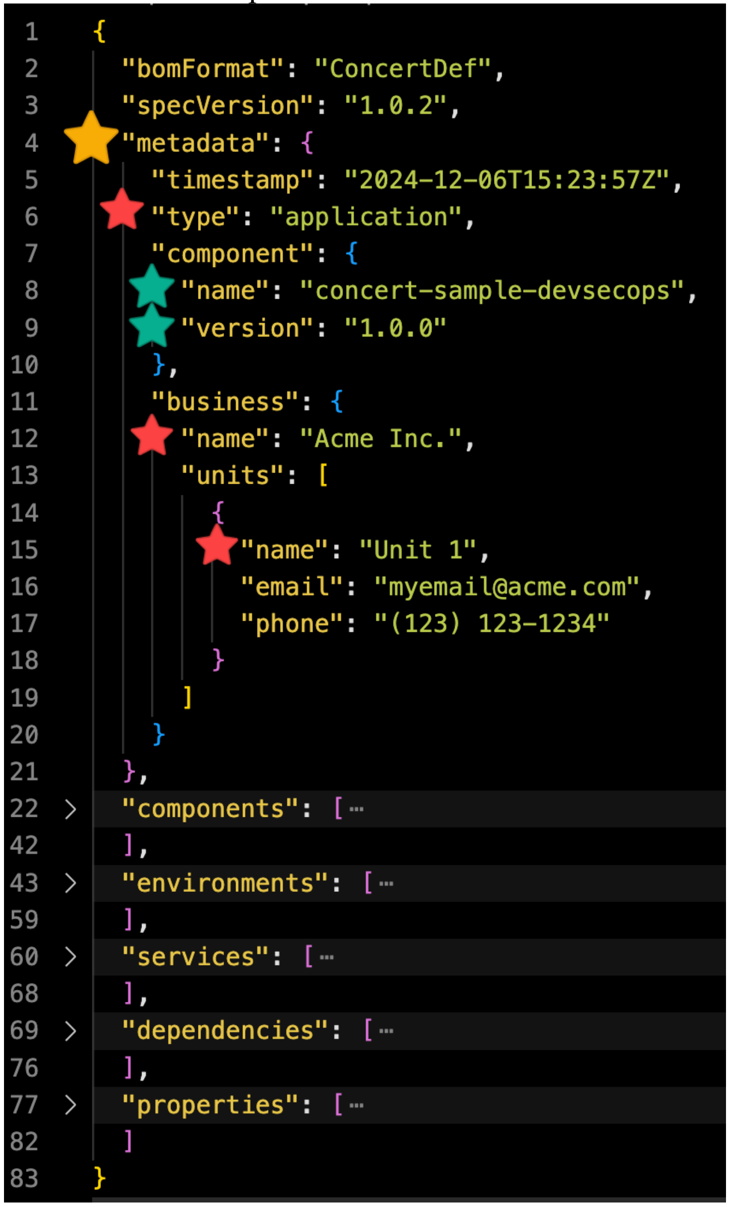
5. ConcertDef Application Blueprint SBOMs

The top-level JSON objects that are specific to ConcertDef *Application SBOMs* are:

- *components*: An array of blueprint objects for build inventories
  - *container*: A structured blueprint data for a container image
  - *code*: A structured blueprint data for a source code repository
  - *library*: A structured blueprint data for a software library
- *environments*: An array of structured blueprint data for deployment environments
- *services*: An array of structured blueprint data for API services
- *dependencies*: An array of dependency relationship objects
  - *ref*: A *bom-ref* which uniquely identify a component or service within the BOM.
  - *dependsOn*: An array of *bom-ref* identifiers of the dependent components or services
- *properties*: An array of name-value objects
- *tags*: An array of strings that can be used for searching or filtering.

Figure 5 shows an incomplete ConcertDef *Application Blueprint SBOM*. Application-specific top-level objects are:

- *metadata*: The *type* field (on line 6) is required, and the value must be "application". 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.

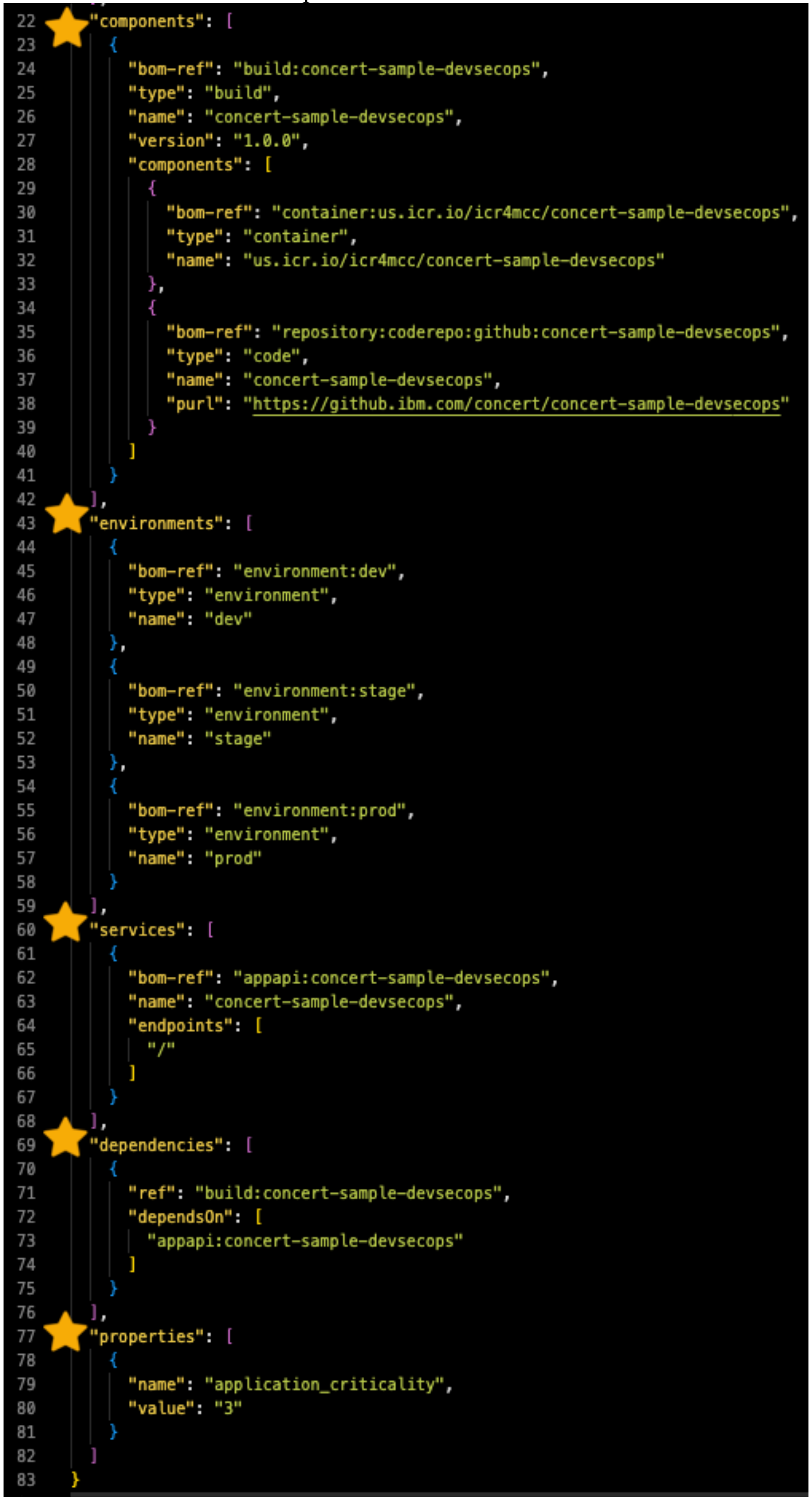


```
1   {
2     "bomFormat": "ConcertDef",
3     "specVersion": "1.0.2",
4     "metadata": {
5       "timestamp": "2024-12-06T15:23:57Z",
6       "type": "application",
7       "component": {
8         "name": "concert-sample-devsecops",
9         "version": "1.0.0"
10      },
11      "business": {
12        "name": "Acme Inc.",
13        "units": [
14          {
15            "name": "Unit 1",
16            "email": "myemail@acme.com",
17            "phone": "(123) 123-1234"
18          }
19        ]
20      }
21    },
22 >  "components": [⋯
42    ],
43 >  "environments": [⋯
59    ],
60 >  "services": [⋯
68    ],
69 >  "dependencies": [⋯
76    ],
77 >  "properties": [⋯
82    ]
83  }
```

Figure 5. An Incomplete ConcertDef Application Blueprint SBOM.

- *components*: An array of the *ConcertDef build objects* selected in terms of the *name* and *version* of the candidate ConcertDef *Build SBOMs*. Contents of the *container* and/or *code* "blueprint" objects in the *Application SBOM* do not have inventory-specific details. Related *Build SBOMs* and *Deploy SBOMs* can be created before or after the *Application SBOM* is uploaded to Concert.
- *environments*: An array of deployment *environment* "blueprint" objects.
- *services*: An array of *service* "blueprint" objects. Note that *name* of the "blueprint" object on line 63 (i.e., "concert-sample-devsecops") is the API service name specified in the *Deploy Inventory SBOM* of the selected ConcertDef *component* (on line 70 in that *Deploy SBOM*).
- *dependencies*: An array of *dependency* objects for the "blueprint" objects in the same file.
- *properties*: An array of application-level properties, e.g., application criticality.

Figure 6 shows the *components*, *environments*, *services*, *dependencies*, and *properties* "blueprint" objects in the sample *Application Blueprint SBOM*.



```
  "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"
        },
        {
          "bom-ref": "repository:coderepo:github:concert-sample-devsecops",
          "type": "code",
          "name": "concert-sample-devsecops",
          "purl": "https://github.ibm.com/concert/concert-sample-devsecops"
        }
      ]
    }
  ],
  "environments": [
    {
      "bom-ref": "environment:dev",
      "type": "environment",
      "name": "dev"
    },
    {
      "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": [
        "/"
      ]
    }
  ],
  "dependencies": [
    {
      "ref": "build:concert-sample-devsecops",
      "dependsOn": [
        "appapi:concert-sample-devsecops"
      ]
    }
  ],
  "properties": [
    {
      "name": "application_criticality",
      "value": "3"
    }
  ]
}
```

Figure 6. Sample "Blueprint" Objects in a ConcertDef Application Blueprint SBOM.

6. Automated Generation & Upload of ConcertDef SBOMs

ConcertDef *Build Inventory* and *Deploy Inventory* data can be automatically gathered and uploaded to Concert with high cost-efficiency via CI/CD (Continuous Integration & Continuous Deployment) pipeline/workflow automation scripts. Such automation can be done with insignificant impact to existing pipeline scripts. For example, IBM *DevSecOps-Concert* provides such automation support at its core such that container-based ConcertDef SBOMs. The built-in capabilities can be exploited easily by configuring pipeline settings and/or by coding simple inventory data gathering scripts. The steps below exemplify how the automation scope of an existing *IBM DevSecOps* CI pipeline can be extended for Concert. Existing CD and CC (Continuous Compliance) pipelines can be extended similarly. The CC pipelines periodically generate vulnerability scan results to Concert under the model of continuous monitoring.

1. Configure the required and optional Concert-specific pipeline settings
2. Optionally, add `concert_deploy` JSON-file creation scripts for *Deploy SBOM* generation
3. Run the CI pipeline and, optionally, confirm the uploads via the Concert Console in use

7. Integration of ConcertDef SBOMs and Other Concert Files

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

- **Code based vulnerability CVEs** are linked with *code* objects in ***Build Inventory SBOMs***.
- **Image based vulnerability CVEs** are linked with the *container* objects in ***Build Inventory SBOMs***.
- **Package SBOMs** sourced from source (or images) are linked with the *code* (or *container*) objects in ***Build Inventory SBOMs***.
- **Certificates** are linked with the HTTPS endpoints (or base URLs) in ***Deploy Inventory SBOMs***.
- **Runtime performance and compliance postures** for Kubernetes-like clusters are linked with the *kubernetes* objects in ***Deploy Inventory SBOMs***.

We note that when a specific code-based package SBOM is uploaded, 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, otherwise only the constituent packages are listed in the GUI. After a specific *Build Inventory SBOM* is uploaded with *BOM-Link* references to package SBOMs, the GUI shows the inventory SBOM's (component) name with each of the referenced package SBOMs.