CMDB APPLICATION MAPPING EXPLAINED

How well an IT function can deliver value to its customers through quality services can be largely attributed to the level of visibility of the underpinning components that facilitate the services to work as well as their dependencies.

While many monitoring solutions in the market claim to provide visibility, the Configuration Management Database (CMDB) has remained the gold standard for a proven capability of visibility at all levels of a service.

Establishing a well working CMDB isn't easy, as Gartner infamously reported 80% failure rates on surveyed CMDB implementations returning business value. But for those who are successful, there is a guaranteed leap in service quality.

Let's look at one of the capabilities—application mapping—that the CMDB delivers at the application level.

What is CMDB application mapping?

ITIL® 4 defines a CMDB as a database used to both:

- Store configuration records throughout their lifecycle
- Maintain the relationships between configuration records

(Understand the differences between configuration items & assets.)

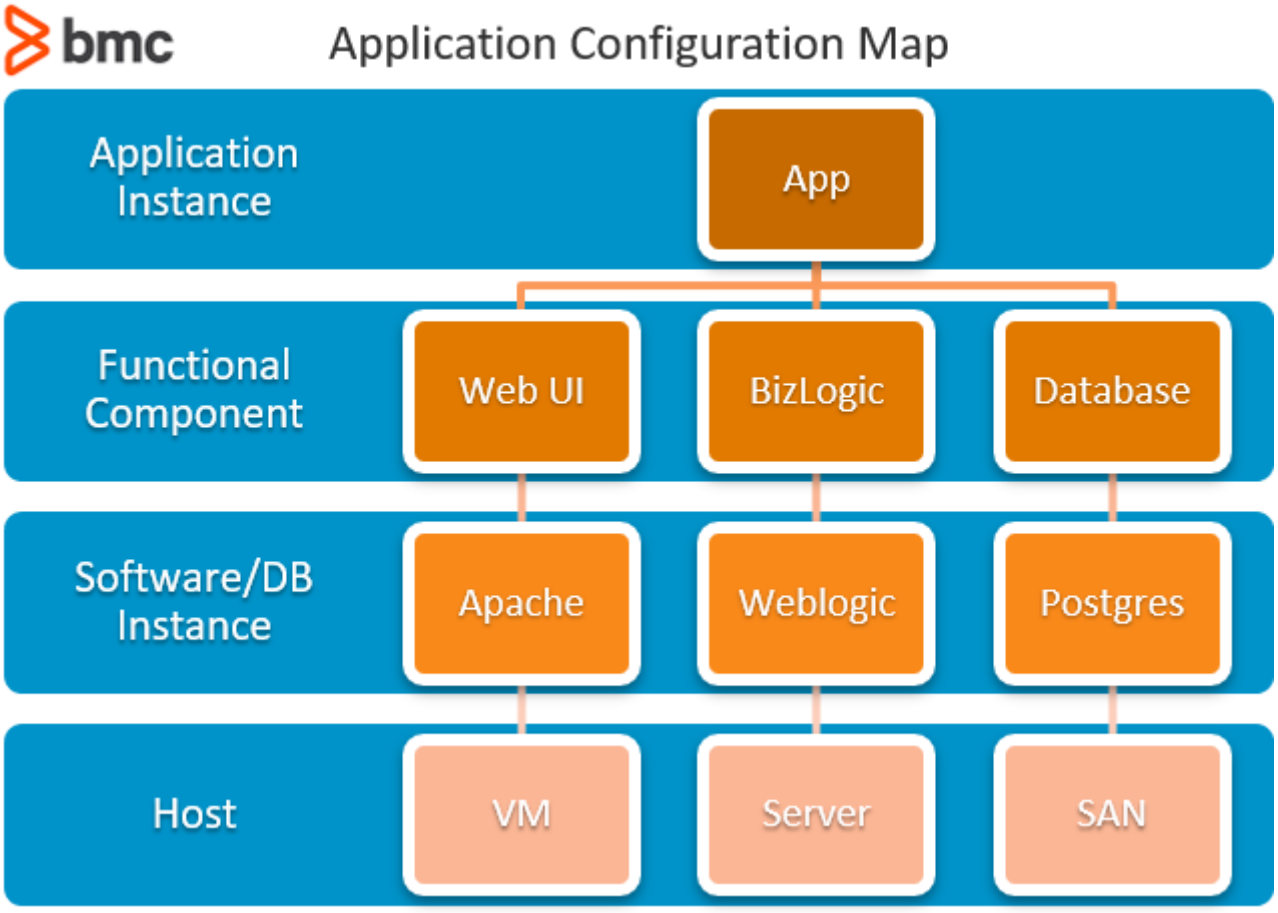
For applications, the kind of configuration records we would seek includes software version, individual installations, and licenses. When it comes to relationships for applications, you would

consider the following components:

- **Physical infrastructure components** such as server and network equipment, e.g. switches, routers, and firewalls.
- **Virtual infrastructure components** including virtual machines, operating systems, hypervisors, middleware, and interfaces.
- **Data components** such as the data and its storage elements including databases and storage area networks.
- **Services** including those that run off the application and those that depend on data to be processed or provided by that application.

Putting all the relevant components together results in a model that can visually depict the components and their relationships—an application map.

The organization of the data usually takes a hierarchical format. But other formats are possible, too: mind maps, data flow diagrams, fish bone diagrams, and similar, as seen in the example below:



(Source)

Approaches in CMDB application mapping

Application mapping was originally carried out as a manual exercise. Tech teams would use static CMDB data and audit exercises to identify and record applications, their underlying components, and their relationships. Then they would switch to mapping software such as Visio to create topology diagrams for their services.

This approach is good for small entities with software hosted in a few locations with few

dependencies. But as complexity and the rate of change within the environment increased due to increasing digitization, manual application mapping becomes untenable.

Today, specialized tools are used to automatically identify and map components based on specified inputs. Automated application mapping is a logical next step after the discovery phase. Here the application signature is identified through:

- The processes and open ports on the application's host VM or server
- The information contained in the application's configuration file such as processes and related elements

Application discovery is also aided by data obtained via Windows (WMI) and Linux (SSH) protocols. This information is obtained through either:

- Polling the components (agentless approach)
- An agent that is installed in the components and sends to the CMDB

The latter option will provide more information, though it does require greater administration effort.

For the mapping portion, the CMDB will extract the discovered information and parse it through a mapping software that will create a topology of the application based on the processes and elements. While the production of the map might be automated, some manual input might be required to specify the scope and arrangement of components, especially for large scale enterprise environments.

Agreement of the topology model is something that needs stakeholder input, since the application map might be useful for a variety of teams and other partners involved in:

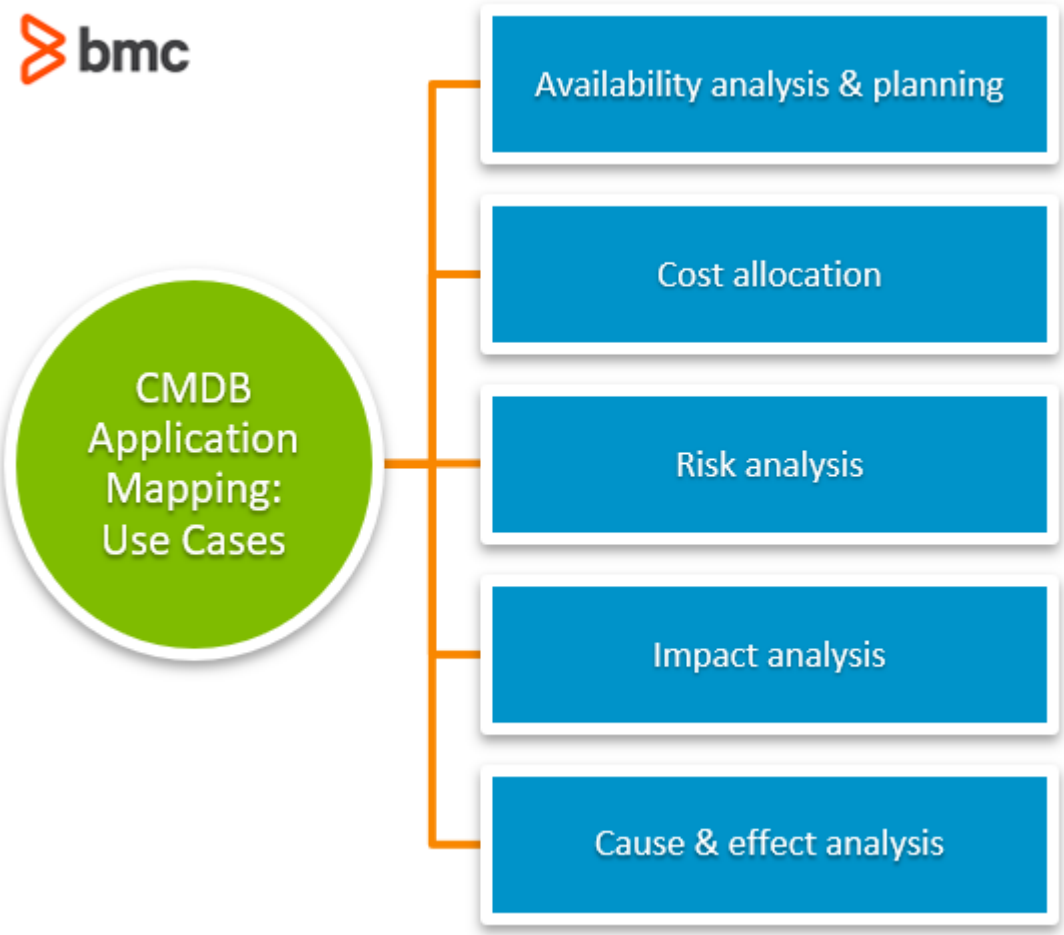
- Architecture
- Development
- Support

Some teams might want a high-level overview for general understanding, while detailed views might be required by others during an investigation or planning exercise.

CMDB functionality should allow for different views to be stored for future reference based on need.

Use cases for CMDB application mapping

The ITIL 4 configuration management practice guide references five purposes of configuration that we can apply to CMDB application mapping:



Availability analysis & planning

Application availability is a key determinant to service quality. The application map serves as a useful input for teams trying to:

- Pinpoint causes of unavailability
- Identify critical components that serve as single point of failures so that relevant availability controls can be designed and deployed accordingly

The information from these exercises can feed into architecture planning, business continuity planning. and other relevant practices.

Cost allocation

The CMDB application map can serve as a key reference in determining and analyzing costs of services and underlying components. Trying to justify improvements or expansion to address demand is better aided through maps that can serve to point out where optimization or spending is required in the service architecture.

Risk analysis

Similar to availability analysis, risk analysis exercises would benefit greatly if all parties involved had a common perspective on the service components as revealed by the CMDB application map. Then it becomes easier to identify vulnerabilities and determine threat factors at all levels that affect

business services and underlying assets.

This then contributes to improved risk treatment planning.

Impact analysis

In the change lifecycle, assessing the impact a change will cause is a critical exercise before authorization and execution takes place. The CMDB application map serves as a critical input for both the teams involved in planning the change and the appointed change reviewers.

While the CMDB application map might not be referenced during a change advisory board (CAB) meeting, the results of the CMDB impact analysis should be considered as evidence in approving a change.

Cause & effect analysis

In the cases of a change gone wrong, an unexpected service outage or experienced degradation, then the questions everyone will ask afterwards about reasons and sources will be better answered when the CMDB application map is used as the common reference point.

CMDB application mapping requires stakeholder support

For these use cases to create value in service management, CMDB application mapping must be viewed as an important exercise that requires the support of all stakeholders. Organizations must surmount the bureaucratic and arduous nature of maintaining CMDB information, by

1. Assigning and enabling required resources for maintenance.
2. Conducting regular audits on the application maps status.

Additionally, consider automation. Set up your CMDB application maps to automatically update or share alerts with configuration owners whenever a change in the environment is detected.

BMC supports CMDBs

BMC Helix CMDB is your single source of reference for assets and services. Want to learn more? View the datasheet.

Related reading

- BMC Service Management Blog
- Introduction To Configuration Management
- Free eBook: Step-by-Step Guide to Building a CMDB
- Service Mapping: How To Create & Use Service Maps
- Value Stream Mapping (VSM) Tutorial with Examples & Tips