e-ServiceSuite™

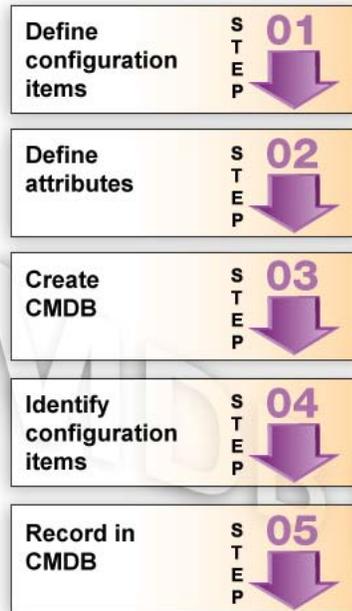# CONFIGURATION MANAGEMENT

Configuration Management (CM)

e-ServiceSuite™ contains a Configuration Management Database (CMDB) that is used to identify Configuration Item records (physical or logical asset records and other items that you wish to track) and define the relationships and dependencies between CIs and their role in the delivery of services

Definition: Configuration Item (CI) is an IT asset or a combination of IT assets that may depend and have relationships with other IT processes. A CI will have attributes which may be hierarchical and relationships that will be assigned by the configuration manager in the CM database.

CI attributes
1. Technical
2. Ownership
3. Relationship

Steps

- STEP 01 — Define configuration items
- STEP 02 — Define attributes
- STEP 03 — Create CMDB
- STEP 04 — Identify configuration items
- STEP 05 — Record in CMDB

Incident Request Process → Update CMDB → Configuration Management Database ← Update CMDB ← Stock Control

Update CMDB ↑

Change Management Process ↑

Activities

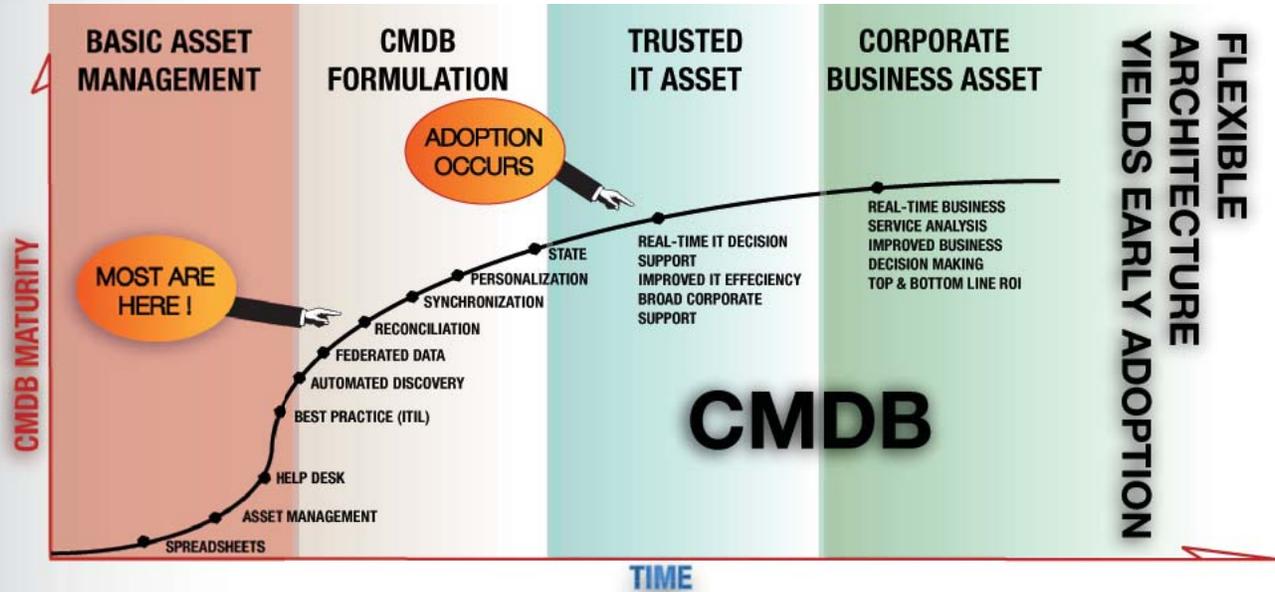The information in the CMDB is used for **five** basic activities:

- Planning
- Identification
- Control
- Monitoring
- Verification
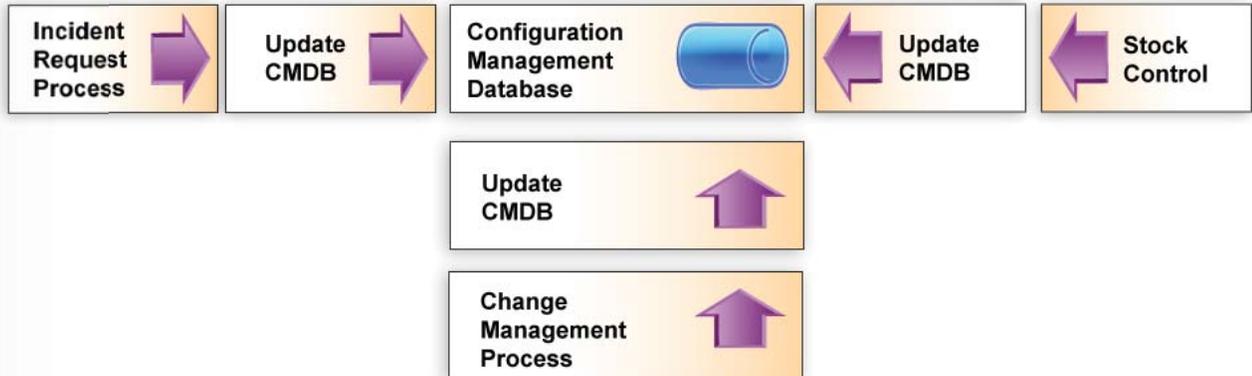
Configuration Management Database (CMDB)

The fundamental component of CM is the CM database (CMDB) which contains the CI information and is used to understand the CI relationships and

Configuration Item Class Definitions

CI Class Definitions

The CI Class of each type of Configuration Item is defined using three levels of categorization as follows:
- CI Class Type
- CI Super-Class
- CI Class

The menus on these fields are hierarchical, so this structure is useful for grouping and to search for similar CI classes. How you use the different levels will vary according to your needs, but some examples are given below:

- You might use the '**CI Class Type**' level to segment IT configuration items from Facilities CIs, or Staff CIs. Alternatively, you could use this top level to segment production from development and test environments, etc.

- The **CI Super-Class** level can be used to group similar types of CIs, so for example you might define a Super-Class of "Computer System" or "Hardware", then the CI Class entries below that might be "Server Hardware", "Desktops", "Laptops" and "Handhelds". This class level is particularly useful when defining allowed relationships between groups of CI classes.

- The **CI Class** level should be used to identify a distinct class of CI objects that share functional attributes such as relevant Status values, the same permission group managing the entries, same extra functionality required (Components, Licenses, Reservations, Finance or End of Life tabs, etc. – see below for further details of this).

Defining Configuration Item Attributes

Multiple records in the CI Attribute table can be created and linked to the main CI record in the Configuration Item form, to record as many attributes of the CI as a required. Each CI Attribute record consists of an attribute name and attribute value pair.

Which attribute data to store for a CI Class can be defined in the Configuration Item Template for that Class, including default values where appropriate, but extra attributes can also be created on each individual CI record if required.

Storing CI attributes this way in a linked table means that if you decide to store extra attributes for your configuration item, no structural changes are required and you simply need to create extra linked attribute records.

Storing Attribute History

By default, a history will be kept of all changes made to attribute values. A copy of the changed values is created automatically in the CI Attribute History form. To turn off this feature for a particular attribute you should check the 'Do not keep history' checkbox. This setting can be made in the Configuration Item Template and will be used for all new CI records based on that template.

Functionality is included to allow you to view attribute history, or to specify a date and view attributes at that date. You may also use this functionality to restore attribute values back to those at a particular date – i.e. to roll back changes to attribute values.

For numeric-type attributes, a history record is only created if the rounded value changes. This is useful for attributes like CPU clock speed where minor variations may be caused by temperature changes – rounding the value stored allows minor fluctuations to be ignored by the attribute history mechanism.

e-ServiceSuite
ITSM SUITE - WORK-FLOW DRIVEN - PROFESSIONALLY DELIVERED - ITIL COMPLIANT

e-ServiceSuite™

powered by:

mx mendix
the app platform

FOR MORE INFORMATION VISIT US ON THE WEB AT:

www.e-ServiceSuite.com
www.e-ServiceSuite.co.uk