**CDB Vocabulary**

* Property is a combination of **property type** and **property value**; examples of **property types** are length, documentation URL, estimated cost, measurement values, etc.
* **Property types** may have any number of assigned **allowed property values**; if no allowed values are assigned to a given type, this indicates that any value is acceptable; otherwise, only values listed as allowed are acceptable for a given type
* **Property values** might change, and hence **property value history** must be kept
* Resource must have **resource type**; it may be provided or used (as either required or optional); examples of resources are power, cooling water, etc.; resource valuesare not constrained and are relatively static (so no history will be kept)
* **Component type** may have **component type category**
* **Component type** may be associated with any number of **property types**
* **Component type** may be associated with any number of **resource types**
* **Component** may be associated with one **component type**; a newly created **component** will have **component properties** for all dynamic **property types** cloned from its **component type**, with **property values** assigned using default **property type** values; similarly, new **component** will be assigned **component resources** that are associated with its **component type**
* Each **component type** may correspond to a generic **component** with the same name
* **Component** may have any number of **component connectors**
* **Component resource** may be assigned **component connector**, in which case connector’s component id must match resource’s component id
* **Component instance** is an instance of a **component**, and must have single physical **location**; **component instance** will have dynamic **property types** cloned from its **component**
* **Component** becomes an assembly (or complex component) if it is assigned one or more children (**assembly components**); in other words, an **assembly component** is a child of another (complex) **component;**
* Distinguishing feature of all **components** (including assemblies) is that they belong to a component catalog
* Two **assembly components** may be connected via **assembly component connection** if they belong to the same assembly
* **Assembly component connection** may designate specific **component connectors**, in which case connector’s component idmust match assembly component’scomponent id
* **Assembly component connection** may also designate separate link **component**
* **Design element** must be associated with a single parent **design**
* **Design element** may also be associated with either **component** or a child **design**; if element specifies child **design** , it cannot specify **component**, and vice versa
* **Design element** may be associated with a **location**
* **Design element** may be associated with a single **component instance**; in this case, **design element** cannot specify child **design**, andmust also specify **component** matching the one that **component instance** belongs to
* **Design** may have any number of **design elements**
* **Design** may have any number of **design properties**
* Two **design elements** may be connected via **design element connection**; the elements do not need to belong to the same **design**
* **Design element connection** may designate specific **component connectors**
* **Design element connection** may also designate separate link **component**
* **Design element connection** may designate **resource type**; if both resource type and component connectors are provided, they must be consistent, and also one of the component connectors must provide resource, while the other one must use it