Next Generation Component Management

Summary

This document provides a conceptual walk-through of Altium's next-generation component management model.

The traditional model of a design component faithfully extends across all aspects of the electronics design process. However, to seamlessly fit the process of electronics design into the encapsulating product development process as a whole, this model needs to evolve – extending to cover other aspects including other design processes, in particular MCAD and Industrial Design, as well as business processes – such as procurement and manufacturing – that intersect with the product development process.

Modeling all these different aspects of the object is fundamental to linking together the different parts of the design process into a single unified whole. Having a single component that models all the different aspects of the object is also critical for enabling a designer to maintain a truly holistic view of the design.

This paper takes a look at how Altium solves this problem through its Unified Component model, and a new approach to component management that harnesses the power of this model.

Historical Component Management in Altium Design Solutions

Altium has been providing innovative, leading edge design solutions for many years. Within the various offerings of the software – from the DOS days to present day with Altium Designer – management of components used in a design has always, and continues to be, a fundamental element of the software. After all, without components, there would be no designs. Without effective management of those components, there would be no ability to reuse them in other, new designs.

If we take a step back to consider the various flavors of component management that have appeared over the years, we can see that the latest component model and management concept is, in fact, the fourth generation:

- **First Generation** – the humble Schematic and PCB library models (`*.SchLib` and `*.PcbLib`) that kicked it all off in the software of the mid-eighties.
- **Second Generation** – the Integrated Component model, implemented through Integrated Libraries. In this model the higher level component is modeled within the schematic symbol in a Schematic library file (`*.SchLib`). Other models are linked from the symbol and component parameters are added to the symbol. All source libraries – symbol and linked models – are defined within a Library Package project (`*.LibPkg`), which is subsequently compiled into a single file; an Integrated Library (`*.IntLib`). The advantage of compiling into an integrated library is that all component information is available in a single portable, and secure file.
- **Third Generation** – the arrival of Database Libraries (`*.DbLib`) and version-controlled SVN Database Libraries (`*.SVNDbLib`). Often termed 'Table-Based Libraries', all information for the components is stored in a format external to Altium Designer such as ODBC, ADO, or an Excel spreadsheet. Each record in the database represents a component, storing all of the parameters along with model links, datasheet references, or other component information. The record can include links to inventory or other corporate component data. As all the detail that makes the component complete is stored in the database itself, the schematic symbol becomes only a graphical representation or, simply, another model.
- **Fourth Generation** – the 'next generation'. Familiar Schematic and PCB library models continue to play an important foundation-level part in this latest generation. However, the component model has now evolved to provide representation not only in the engineering domain for the designer, but also in the wider product design arena accessed by the procurement and manufacturing teams. A 'Unified Component' model where file-based definition of components is performed on the design side, and then released into a secure storage repository, or vault. Here, a component is stored as a series of revisions of a uniquely-identifiable component Item. Each revision is lifecycle-managed, providing collections of certified components, authorized to be re-instantiated into new design projects, manufactured into prototypes, or used for production runs. In short, a catalog of Unified Components implemented through vault-based libraries.

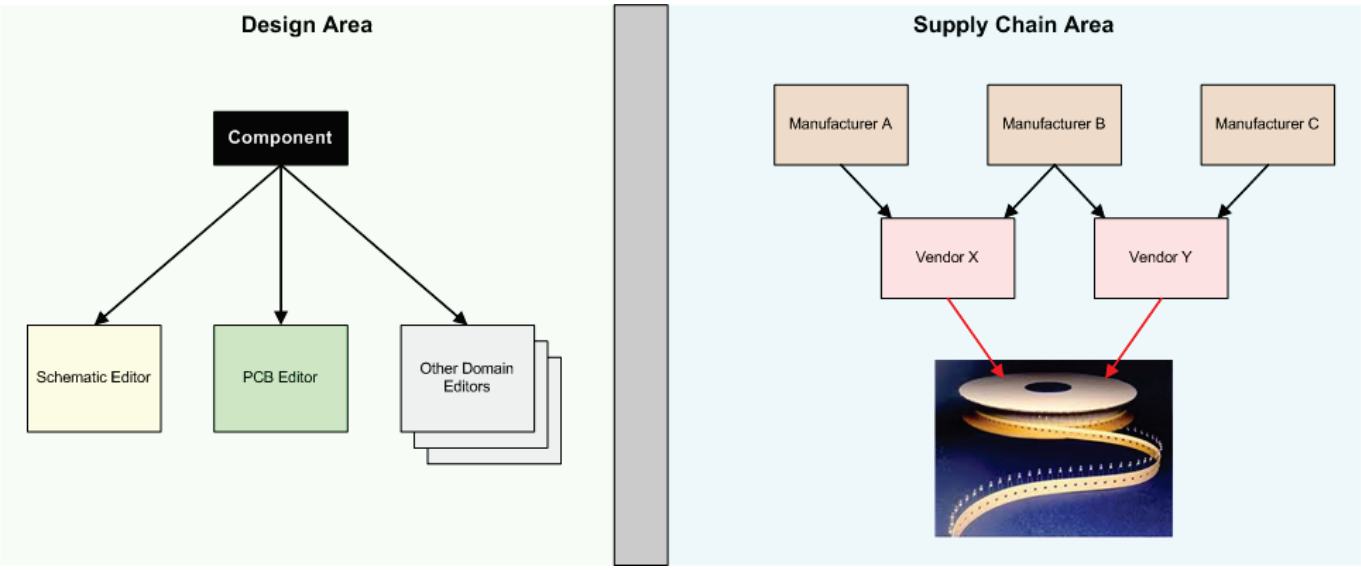
A Component by any other name...

The term 'component' can have different meanings, depending on your perspective. The component concept as seen by the designer is quite different from the component concept as seen by the procurement team.

In the Design Area, the component is a logical entity that is used by the designer to solve problems or implement features. In the schematic editing domain, it has one graphical representation. In the PCB editing domain, another. In still more design domains, it will be represented again in a different shape or form. But at the end of the day, to a designer the domain is irrelevant, the component...is the component...is the component.

So the designer essentially works with a black-box entity that must possess certain characteristics, typically declared by the designer in parametric form and associated to that entity – "Here's a component and this is what it should provide or give me".

For a procurement team, knowledge of the functionality of the component in terms of the wider design is not required. The procurement specialist simply needs to know what real-world physical component they should order. And the procurement specialist has to deal with both Manufacturer Items, as well as Vendor Items – the former being the physically manufactured entity, while the latter is the purchasable entity. In some cases, the Manufacturer of a component may also be the vendor who sells it!



Initial The component seen from two different perspectives. Typically a flurry of external communications is required to relay to the procurement team what is required in terms of implementation of a design-side component in the real world, using manufactured parts sourced from suppliers (vendors) of those parts. To all intents and purposes, these two areas are separated by a virtual 'brick wall'.

So, for example, the designer specifies a resistor with specific properties – a resistance of 10K, in 0405 package, with 5% tolerance. These are the parameters needed to 'fit the bill' as it were for that component when operating in the greater circuit of the designed product. This may map to a reel of 5000 Panasonic resistors that is purchased from Digi-Key for a certain price. Or a reel of 10,000 Yageo resistors that is purchased from Mouser.

Complicating this even further, the design engineer will need to say which of these is OK to "fill the slot" wherever this design component is used. This may include resistors with 1% tolerance, even though the design component is a 5% resistor. And ultimately, the procurement team may opt not to buy resistor reels from a certain supplier because they have found their delivery is unreliable.

So the designer's view of the component really does need to be quite different from that of the person responsible for procurement of that component. But the two worlds do not need to stay in a state of 'splendid isolation'. To do so can prove time consuming and costly. Picture the designer just making that all-important deadline, only to find an email from the procurement team to say that the part required is no longer available, or has a lead-time of several weeks – weeks that may prove seriously detrimental to the company's plans for releasing to market!

If only there were some way to provide the designer with a better 'up-front' picture – and choice – of which physical manufactured items to use to implement specified design components when assembling the board...

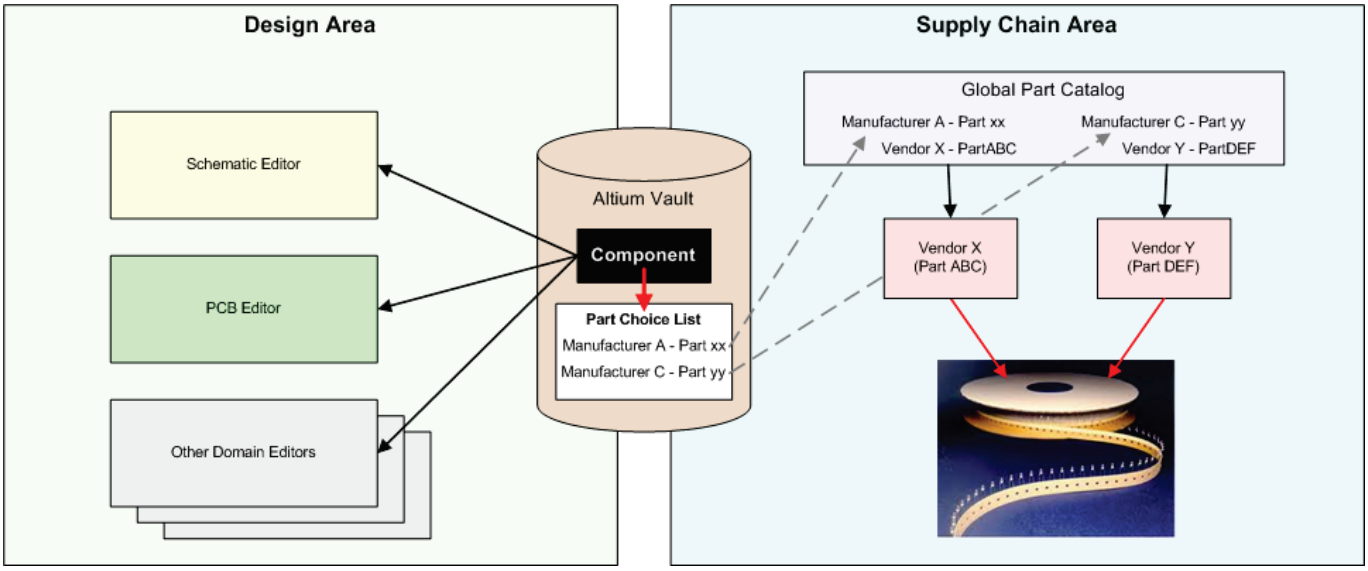
The Unified Component Model

Altium's next-generation component model effectively maps the concept of a design component – in the traditional electronics design arena – to the component as seen by the rest of the organization in the bigger 'product arena'. A truly 'Unified Component' model that not only represents the component in the different design domains (Schematic Capture, 2D/3D PCB Layout, Simulation, Signal Integrity) but also facilitates choices of the desired physical components – real-world manufactured items – at design-time, offering a significant improvement in terms of procurement cost and time, when manufacturing the assembled product.

Under this modeling paradigm, the design component, as seen by the designer, is separated from the manufacturer and/or vendor parts. This information is not defined as part of the component. Instead, a separate document – a *Part Choice List* – is used to map the design component to one or more manufacturer parts, listed in a *Global Part Catalog*, which in turn can be mapped to one or more vendor parts, allowing the designer to state up-front, what real parts can be used for any given design component used in a design.

And the components themselves, along with their part choices, are stored in a centralized repository – an *Altium Vault* – that effectively provides the common ground between the two sides, the 'connecting door' through that virtual brick wall!

The vault facilitates the secure handling of data with high integrity, while providing both designer and supply chain access to those data as needed. No external liaising through email or phone calls. The information required is specified for both sides to see and act upon.



The term 'Unified Component' depicts the extension of a design component into the bigger product arena through dedicated part choices, that map that design component to the real-world manufactured parts that have to be sourced by the procurement team. This gives the procurement team a beneficial and timely 'heads-up' on what those parts should be.

The following sections take a closer look at the different elements that collectively facilitate the implementation of this new modeling paradigm, and the benefits it brings.

The Altium Vault

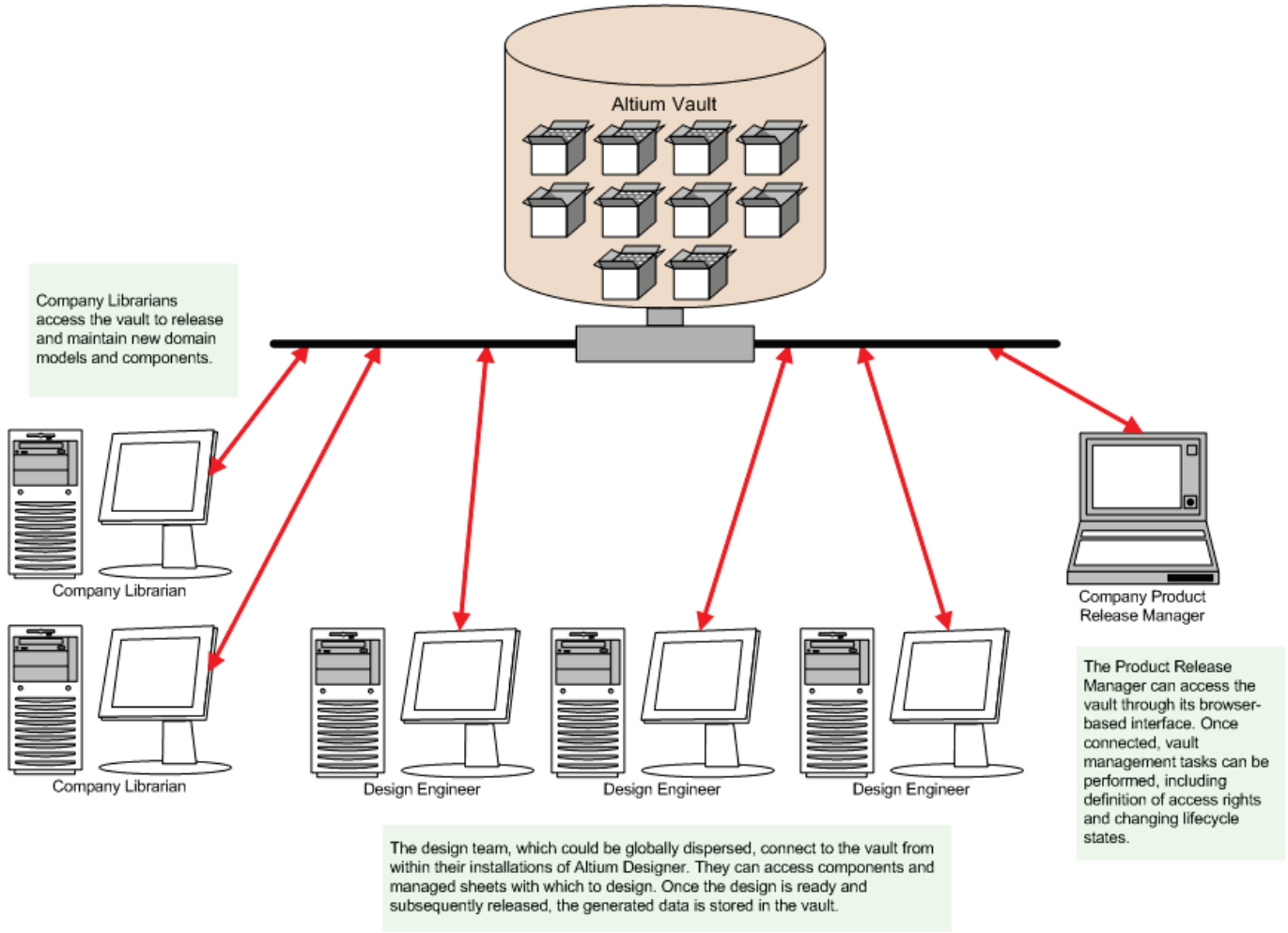
Before looking at how the components themselves are defined, it is prudent at this stage to consider where exactly they, and their associated Part Choice Lists, actually reside.

Under the umbrella of Altium's Design Data Management System, the landscape is essentially divided into two – the *Design Area* and the *Supply Chain Area* – analogous to two neighbors on either side of a garden fence. On the design side, we have a team of designers (sometimes even a one-man show!) that create something innovative. A product that will, in some way, enrich the lives of the people that purchase and use it. On the supply chain side, we have all the teams that are collectively responsible for turning a design into a physical product. This includes, fabrication, procurement, assembly, testing, and so on.

The supply chain acts on, or uses, data that come, for the most part, from the design area. These data are said to be 'released' from the design area. But what actually happens to these released data? How are they handled and what's more, is that handling secure? In other words, can the integrity of the data be guaranteed?

There needs to be an effective, efficient means by which to handle such data in a highly secure, yet readily-accessible manner. A solution that not only provides rock-solid storage of the data, but also enables re-release of the data as distinctly separate revisions – essentially tracking design changes over time, without overwriting any previously released data. A solution that also caters for the lifecycle of the data to be managed, allowing those that need to use the data to see, at-a-glance, what stage they have reached in their lifecycle and therefore what they can be safely used for.

Catering for all these factors, Altium provides a server-based engineering content management system solution –- an *Altium Vault Server*. As a distinct design solution in its own right, an Altium Vault works in harmony with Altium Designer to provide an elegant answer to the question of secure management of data.



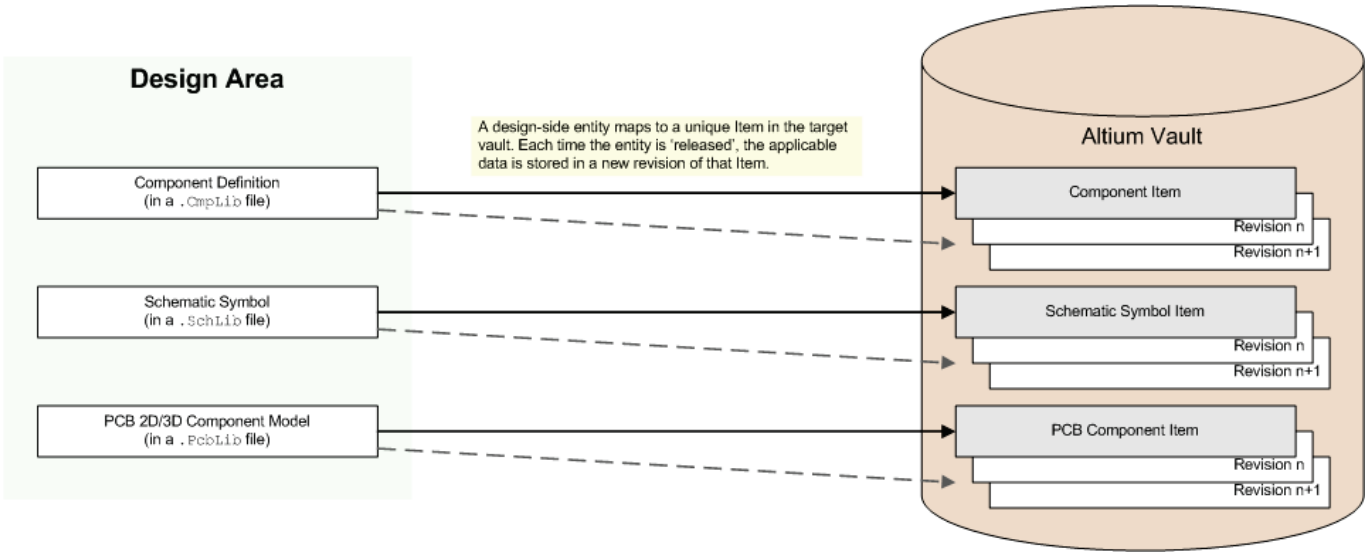
The concept of a LAN-based Altium Vault Server.

In fact, an Altium Vault is used to store more than just data that are released from the Design Area. It is also used to handle other data, data that could originate from the Supply Chain Area too. Part Choice Lists are a good example of data that are very much vault-based in nature, but are not sourced from the design side, and are therefore not 'released' entities.

The various data entities (released from the Design Area or otherwise) are represented in an Altium Vault by unique Items. An Item simply represents a specific object, and is uniquely identified through use of an assigned Item ID. An Altium Vault then, is a centralized storage system into which all data, for each Item, are stored.

What exactly is represented by, and stored in an Item can vary, depending on what has been mapped to that Item. In terms of component management, there will be dedicated types of Items to store the data for not only the components themselves, but also the domain models that they reference.

In a vault, each Item is stored as a series of revisions. Each revision contains data that are used to represent, or to build a particular version of that Item. Each time a change is made to the source design data, a new revision of that Item is created in the vault, ready to accept (store) the generated data.



The concept of Item Revisions, illustrated for the release of a component and its referenced domain models.

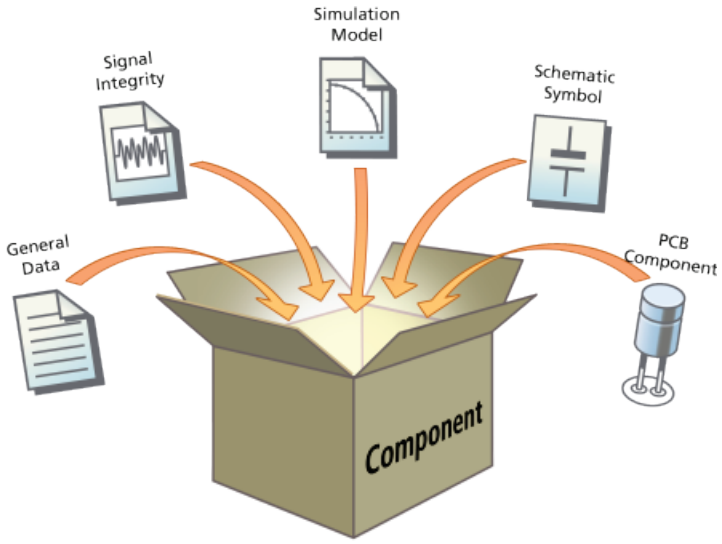
Design Component Definition

Before we take a look at how a design component, through part choices, is mapped into the real world of manufactured parts – becoming an expanded 'Unified Component' – let's take a look at how it is actually defined and managed.

The engineering, or design view of a component is, in essence, a container into which all information used to model that component in the Design Area is stored. This includes links to all requisite domain models (schematic symbol, PCB 2D/3D component, SI, Sim, etc), as well as parametric information. What's more, this abstracted 'bucket'-type modeling is very scalable, should the component need to be represented in additional design domains in the future.

But how to specify a component? Or, more precisely, how to determine the contents of that container? The answer, on the design-side, is to create a *Component Definition*.

A component definition is simply just that – a definition of a particular design component. A definition that ties together the required models and parameters for that component in a clean and ordered fashion. Each component definition on the design-side maps to an Item – a Component Item – on the release-side. To put this another way, you are defining the source definitions that will, when released, provide a set of components which you can re-use again and again in your designs.

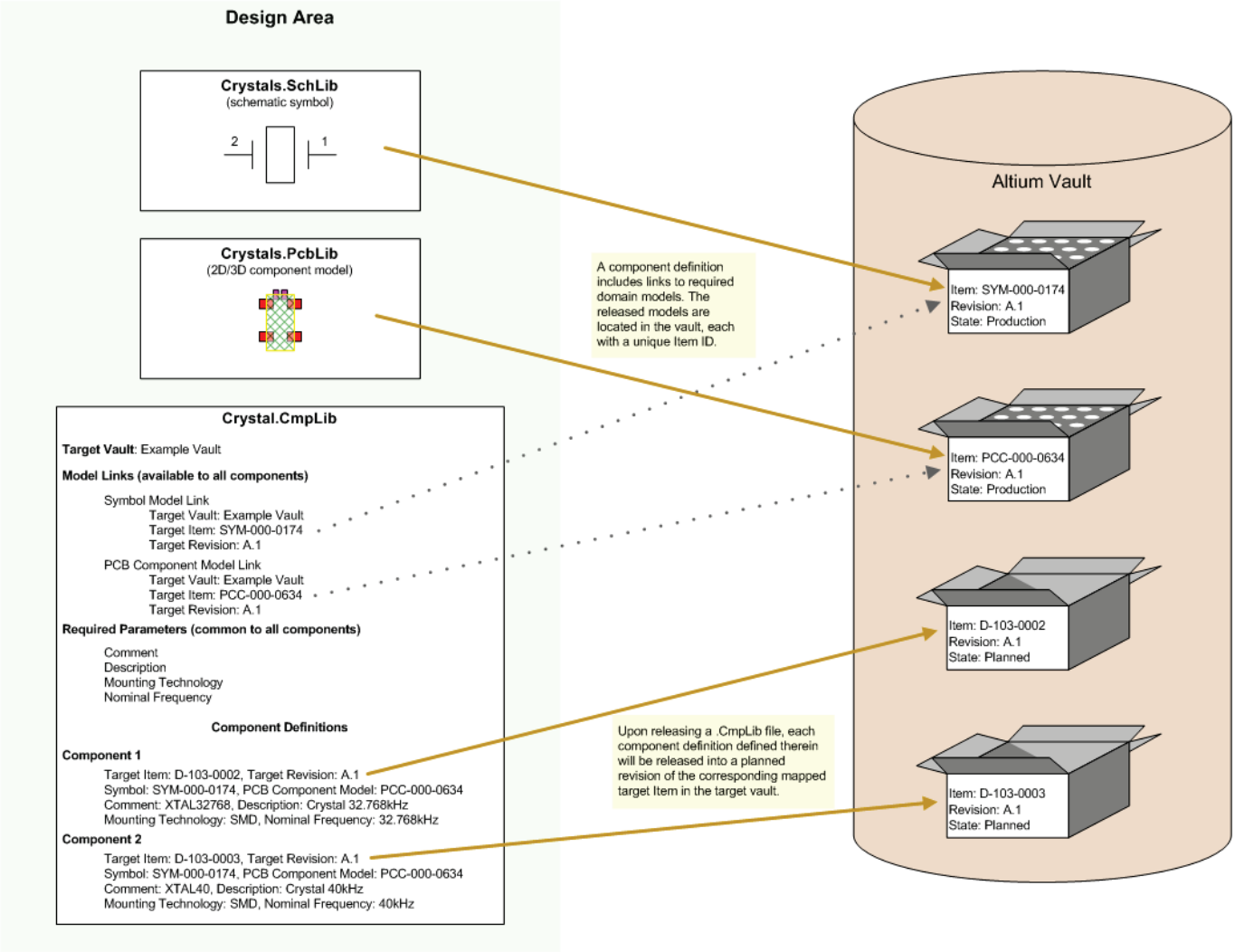


The design view of a component – a neatly packaged container for all information required to represent that model across the various design domains.

In terms of storage, component definitions are created and managed within a dedicated Component Library file (`*.CmpLib`). A single Component Library file can be used to create (and therefore map to) one or more unique Component Items, by entering one or more component definitions. Each component definition will have a common set of parameters and links to required domain models.

> In general it is good practice to have one component definition per Component Library but there are exceptions where it makes sense to manage components as a set, such as a set of chip resistors for example. Component Libraries provide for hierarchical factorization of models so when there are large sets of components that share symbols or footprints then sharing these in a single Component Library can facilitate a higher level of data integrity. If a footprint is changed for example, all of the 0603 chip resistors can be easily updated to use the new revision of that footprint, without the risk of missing one.

Having defined the component definition(s) as required in the Component Library file, you simply release it. The release process is simply the act of generating a new revision of each targeted Item. What's more, the Items are created on-the-fly as part of the release process – define-and-go as it were.



Definition of a component. All data required for a component are specified on the design-side within a dedicated Component Library file, including links to the lower-level released domain models and parametric information.

Domain Models

In terms of its representation in the various domains, a component doesn't contain the domain models themselves, but rather links to these models. These links are specified on the design-side, as part of the source component definition – from which the released component is generated. As such, before you can delve into the process of defining and releasing components, you must first ensure that all the domain models themselves have been created and released.

- **Schematic Symbol** – a drawn schematic symbol within a Schematic Library document (`*.SchLib`) on the design side is mapped to a Schematic Symbol Item in an Altium Vault. Each release of the library stores the symbol model data into a new revision of that Item. Create a new Schematic Library file and add/draw symbols as required, or use an existing library. A single library may contain any number of schematic symbols, with each independently linked to a unique Item in the target vault. One important thing to observe here is that you are creating purely a schematic symbol – the representation of the higher-level component within the schematic editing domain. It is not a 'schematic component' as defined for use in integrated libraries, where other models and parameters are defined as part of that schematic component. The higher-level

If the symbols are coming from old library sources, you do not need to delete any linked model and parameter entries. The system will simply strip these during the process of releasing the symbols to the target vault. This 'stripping' applies to the released data – the design-side source schematic component definitions are not altered in any way, allowing you to still use these same libraries with other component and library management methodologies supported by Altium Designer.

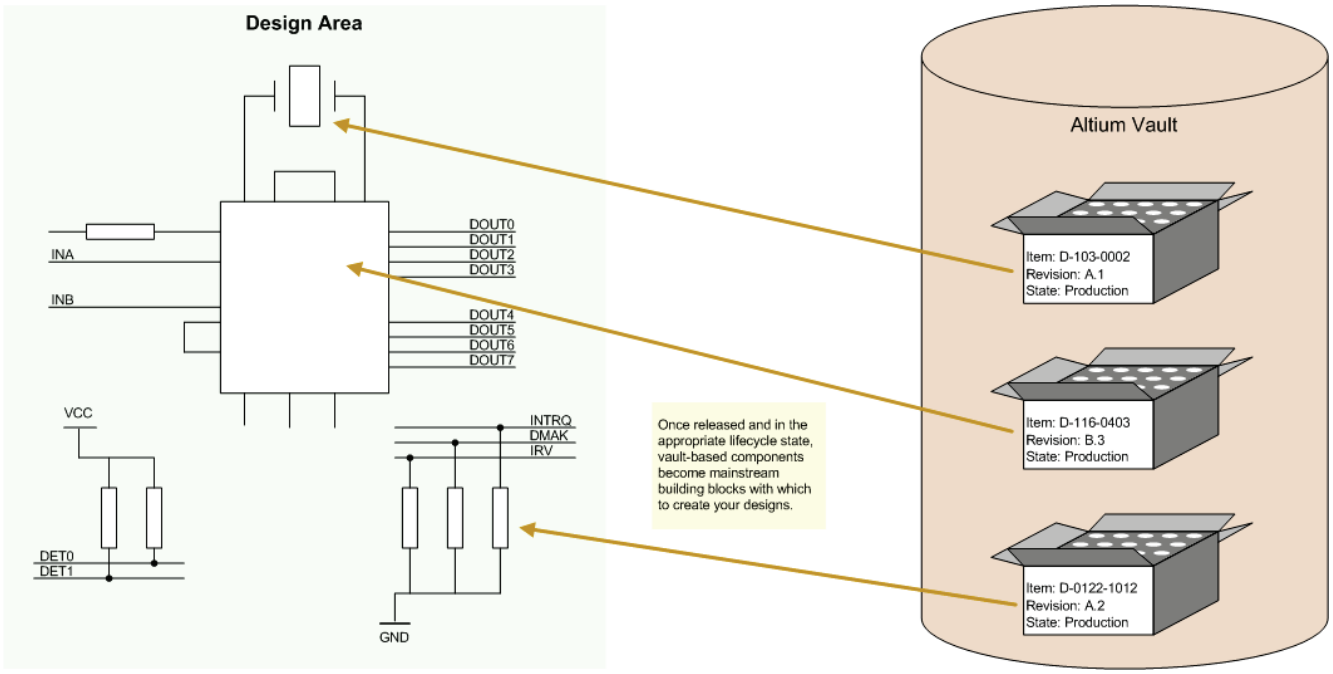
design component needs only the graphical depiction of the symbol. It will include links to other domain models and parameters as part of its own definition within a Component Library file.

- **PCB Component Model** – a PCB 2D/3D component model within a PCB Library document (`*.PcbLib`) on the design side is mapped to a PCB Component Item in an Altium Vault. Each release of the library stores the model data into a new revision of that Item. Create a new PCB Library file and add/draw the 2D footprints as required. Add also additional 3D body information to each footprint if applicable. A library may contain any number of PCB 2D/3D component models, with each independently linked to a unique Item in the target vault.
- **Simulation Model** – a simulation model definition within a Simulation Model file (`*.SimModel`) on the design side is mapped to a Simulation Model Item in an Altium Vault. Each release of the file stores the model data into a new revision of that Item. The released data consists of the model definition in the `.SimModel` file, as well as any referenced `.mdl` or `.ckt` file. For a modeled digital component, there will be the intermediate `.mdl` file, as well as the referenced Digital SimCode `.scb` file.
- **Signal Integrity Model** – a signal integrity model definition within a Signal Integrity Model file (`*.SiModel`) on the design side is mapped to a Signal Integrity Model Item in an Altium Vault. Each release of the file stores the model data into a new revision of that Item. The released data consists of the model definition in the `.SiModel` file, as well as pin model definitions stored in a `.SiLib` file.

With respect to SCHLIBs and PCBLIBs, although a single library can contain multiple symbols/models, from a version control perspective it is best practice to have one symbol/model per library file. This allows you to check out and modify just the symbols/models you need to modify, without registering a version change to an entire, single source. The system provides a tool to split schematic and PCB source libraries into individual libraries, each containing a single schematic symbol or 2D/3D component model. The system also provides the ability to batch-release, in-turn, source SCHLIBs and PCBLIBs.

Certified Components for use in Designs

Once released and available in the vault, components – or *Vault-Based Components*, if you like – can be re-instantiated into any new design project, manufactured into prototypes, or used for production runs. In addition, the concept of component certification is made possible as the components are formally revised and lifecycle-managed. This allows the organization to specify the state of its components and what they can be used for (design, prototype, production, etc). From a design perspective, this results in the creation of vault-based libraries, containing a formal collection of components that have been company-approved for use in each new design project embarked upon within that company.



Design using components that have been certified for use!

The beauty of using certified components in your designs is that when it becomes time to change the lifecycle state of your board design, the integrity of the design becomes greater still, since a design can only be released to "Prototype" or "Production" provided the components it uses are also in a corresponding state. Put another way, you wouldn't start to produce that assembled board if the components are only at a "Design" stage!

And, if we take this to the finest level of granularity in the component management arena itself, the system will not allow you to promote the lifecycle state of a component in the vault until all of its referenced domain models are in a corresponding correct state to be able to do so. In other words, a parent component cannot be further in its lifecycle than its child models.

Choosing Manufactured Items at Design-Time

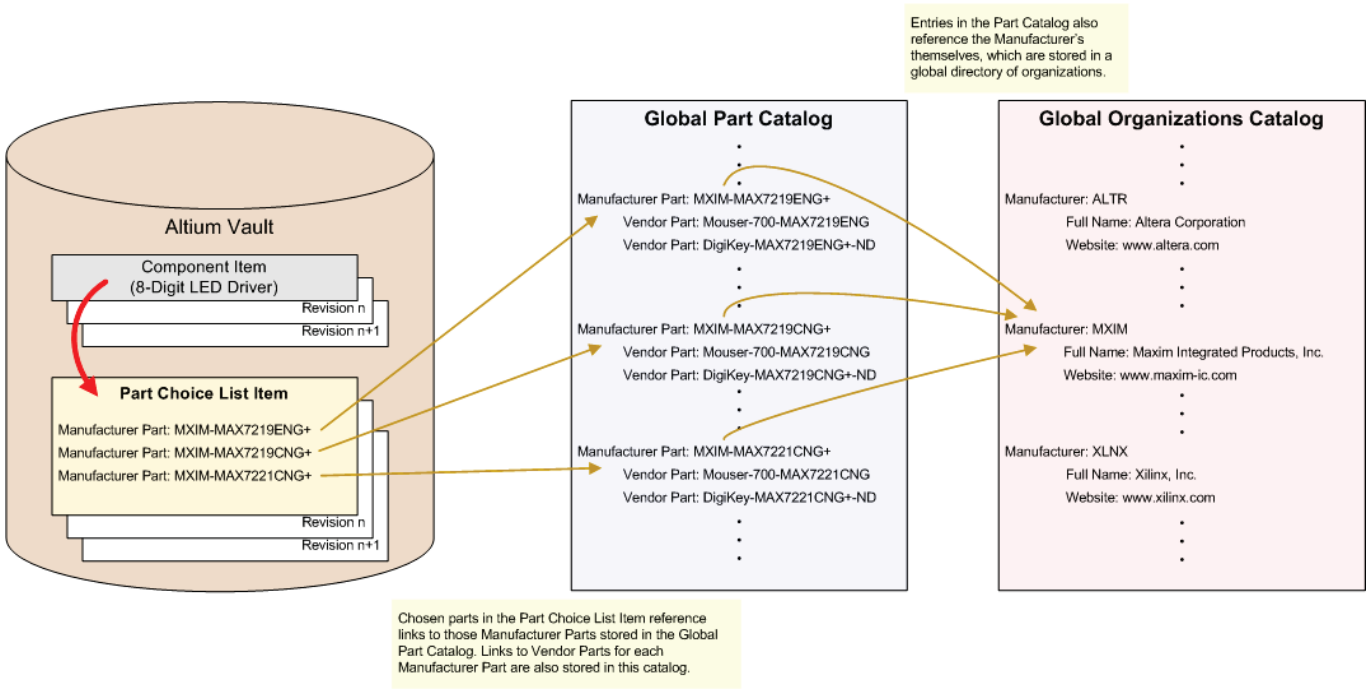
As a board designer, you capture the idea for that next great product using a collection of logically wired component Items across a number of schematic sheets. The component Items you use will have defined symbols (and other domain models) and some key parametric data, but at the end of the day they are just 'symbols of design intent' – having meaning in the context of the project design arena, but holding no physical meaning outside of that arena. Each of these Items needs to be 'embodied in the real world' either by purchasing an off-the-shelf (OTS) pre-manufactured item, or by having it made to spec (MTS).

Recapping, in the Supply Chain Area, a procurement specialist often does not know (or even need or wish to know!) what a particular component Item in the design represents. What is required, is an indication of what needs to be procured – which physically-manufactured components can be used to implement that component Item. The best person to indicate which real-world components can be validly used to implement the design-level components would be none other than... the designer.

Wouldn't it be great if, as part of releasing and managing company-ratified design components, information pertaining to the 'allowed' manufactured parts – to implement those components on the manufactured and assembled boards – could also be specified? Well, as part of the Unified Component modeling paradigm, Altium caters for exactly that through the provision of a *Global Part Catalog* and the concept of making *Part Choices*.

The Global Part Catalog is a managed reference catalog with links to off-the-shelf, purchasable components that can be used in the construction of electronic systems – *Manufacturer Parts*. Each of these Manufacturer Parts is linked to the corresponding Manufacturer in a parent *Global Organizations Directory*. In addition, each Manufacturer Part includes links to *Vendor Parts* – those parts that are available to be purchased by a selling organization. In some cases, the Manufacturer may also be the direct Vendor/Supplier.

Mapping itself – from a Component Item in an Altium Vault, to nominated Manufacturer Parts in the Global Part Catalog – is performed using a dedicated Part Choice List Item, which itself is stored in the vault. Each Component Item references its own Part Choice List Item. The revisions of that Component Item will utilize that same Part Choice List Item. The Part Choice List Item itself is revisable and enjoys simplistic lifecycle management. And by keeping this as a separate entity, it can be updated independently of the Component Item that references it.



Mapping a vault-based component to real-world manufactured parts through the concept of Part Choices.

The designer can feel truly empowered by being able to specify Manufacturer Parts that are truly interchangeable at manufacturing time in the context of any usage of that component in their design – the very essence of true part equivalency. And it is this intelligent mapping of a component, that turns the humble vault-based component into a truly Unified Component!

Real-Time Supply Chain Information

The unified nature of a vault-based component, through the chosen part choices made for it, ultimately create a link from that component, all the way through chosen manufacturer item(s), and on to the supplier (vendor) items that each itself references. From the designer's perspective, the component is hooked directly into the supply chain. And this creates exciting possibilities – possibilities that Altium does not overlook!

Real-time data are made available – fed back from the supplier's web services – to let the designer know the current costing and availability of the chosen parts, and from all vendors that sell those chosen parts (as defined in the Global Part Catalog). And not just the designer gets to see this information. The procurement specialist can also keep abreast of supply-chain information, as it is made available in the vault for each component Item therein.

A part is no longer available or has suddenly become cost-ineffective? No matter, provision is made for real-time updates to be sent back to the Design Area as soon as a change occurs. With this vital 'heads-up', the designer can take that choice of part out of the associated Part Choices list for that component and essentially 'off the radar'. And at any time new, truly equivalent parts can be added to the list, should something more appropriate, available and cost-effective come along.

Powerful Where-Used Capabilities

The highly relational structure of the released data in an Altium Vault lends itself to powerful 'where used' capabilities. At any time, you are able to see where a particular child Item is used, in terms of parent Items in the vault. So for a given domain model, you can quickly identify which component Items reference it. For a given component Item, you can see which designs it has been used in, which managed schematic sheets, and so on.

If a component is deprecated or made obsolete for any reason, you can quickly identify and pull-up the schematic sheets and board-level designs in which it is placed, and update and re-release those Items as new revisions using another, equivalent approved component.

And why re-invent that proverbial wheel, when you can find a component that you need to use and see which designs it has already been used in. If those designs are at a production state, that component has been proven already and is therefore good-to-go. It can be re-used in your next design with confidence.

A Cornerstone of a Grander System

This next-generation of component management is, in itself, a powerful system. Yet it is part of a greater system still. Along with managed design reuse of schematic sheets and the high-integrity release of board designs to manufacture, it makes up the set of mutually-beneficial, yet independently-functional sub-systems that together form Altium’s wider Design Data Management System.

And as these other systems rely heavily on the availability and use of components, the ability to define released collections of secure, certified components that can be tracked not only in terms of revision and lifecycle state, but also in terms of where they are used becomes a highly-motivating factor in the decision to migrate to the use of vault-based components.

You are in no way forced to migrate however. You can step into the water gradually, instead of diving straight in – utilizing perhaps the secure, high-integrity storage aspect of these components. Or perhaps tinkering with lifecycle management. Or even enjoying the direct connectivity and access to supply chain information. The existing component management methodologies continue to live on side-by-side with the new system. The point is, Altium provides the tools so that when you are ready, and confident in your knowledge and operation of the system, you can easily migrate your existing libraries of components to create new vault-based libraries of components. And once you embrace the next generation of component management, you'll not want to look back.

Altium’s next generation component management is an evolving entity. In its newly-conceived embryonic state, it provides the essential foundation from which to grow and mature into a far more significant system. A system that paves the way for true web-based manufacturing. A system that affords the ability to share and trade in design IP. And, looking to the future, the eventual arrival of field-upgradeable devices from FPGA and Embedded IP projects that are integrated into parent board-level designs. True cloud-based device management, supporting a globally-connected eco-system of smart devices.