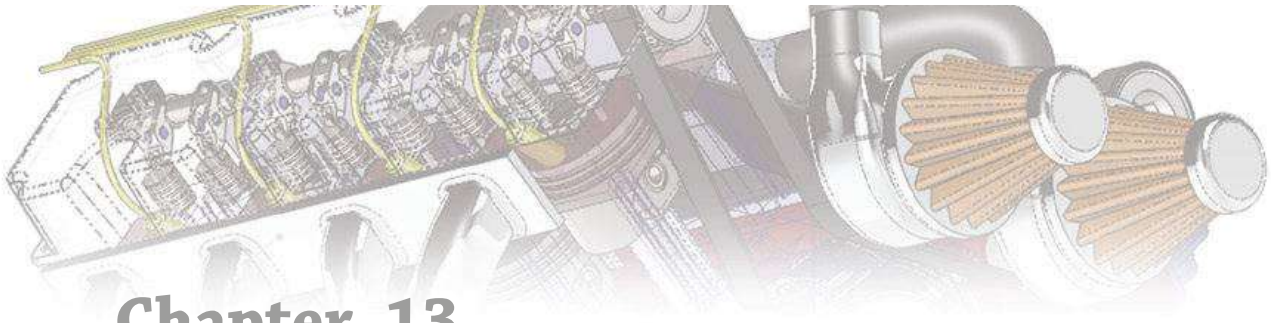


Chapter 13

Building Efficient Assemblies

This chapter introduces you to techniques you can use to manage performance issues as well as general-use issues, efficiency, browse-worthiness, and searchability in assemblies.

IN THIS CHAPTER, YOU WILL LEARN TO:

- Set apart the elements of an assembly
- Increase performance by using SpeedPaks
- Organize assemblies by using subassemblies
- Group parts and mates by using folders
- Show names and descriptions with Tree Display options
- Employ helpful assembly tools

Understanding the Purpose of Assemblies

In the physical world, assemblies exist for several reasons:

- Separating materials
- Allowing relative motion
- Reducing material
- Allowing for different manufacturing techniques
- Allowing for disassembly or repair

Independent of the reasons stemming from physical-world requirements, CAD assemblies might have some unique reasons for existing:

- Depicting an assembly process such as order of operations
- Specifying dimensional assembly relationships and tolerances
- Establishing clearances and limits of motion
- Visualizing motion and spatial relationships between parts
- Designing parts in-context

Mastering SolidWorks, First Edition. Matt Lombard.

- Creating a parts list for assembly (Bill of Material, or BOM)
- Creating a parts list for purchasing
- Automating data entry through product data management (PDM)
- Staging renderings
- Creating data for downstream applications such as animation or motion analysis

You can probably come up with a number of additional reasons for making CAD assemblies. In fact, almost as many reasons exist for making assemblies as there are people making those assemblies.

If you are trying to drive product development with a single top-level assembly, you might run into situations where the various functions of the assembly start conflicting with one another. For example, you might have an assembly where a part flexes. It is difficult or impossible to make flexible parts work effectively in SolidWorks with dynamic assembly motion. Another situation might be in-context relationships where the parent and child components move relative to one another—or maybe you need an assembly for a rendering and the assembly must have multiple instances of in-context components, which can be tricky to manage. You get the picture. You can't always do everything with a single assembly.

You can certainly have multiple assembly files for a single product. In fact, in some cases, this may be necessary. Rendering is probably one of the most common reasons to create a new assembly. Conflicts between external references and motion are another common reason to create a new assembly document.

Identifying Types of Assemblies

The average SolidWorks user thinks an assembly is a collection of parts put together with mates that position parts and may also allow motion. In this kind of assembly, you might use patterns, configurations, in-context techniques, and so on. The goal of the assembly is probably to simulate reality in the way the product looks and moves.

Driving an Assembly with a Base Part and Mates (Bottom-Up with Mates)

This is considered "orthodox" SolidWorks assembly usage and is the way the SolidWorks training materials describe creating assemblies. Insert a part or subassembly at the origin, which becomes fixed in place automatically, and then start mating parts and subassemblies to the base component and add on from there.

It's difficult to criticize an assembly modeling method that has been used for so long by so many people, but the high failure rate of mates attached to edges and faces speaks for itself. SolidWorks has tried to solve this problem for many releases of the software, but the failure rate hasn't changed much, if at all. You may find that the software even tries to hide certain types of mate errors to make them easier to ignore.

If you read Chapter 12, "Editing, Evaluating, and Troubleshooting," you know that the most common methods for part modeling are also the easiest and most error prone. I ask you to

consider that the same is true in assembly modeling. To make truly robust assemblies, a little more forethought is required. In Chapter 12, we used reference geometry to build the bones of the part because that is a more stable approach. Here with assemblies I will recommend the same.

The bottom line for the method of mating to base parts is that it is unreliable through changes. Of the methods that are presented in this book, this is the most common yet least reliable method. This is not the fault of the software, but of the method. This faulty method is popular because it's the easiest and requires the least planning.

An example of where you might use this kind of assembly is a robotic arm. Figure 13.1 shows an assembly that was created with bottom-up techniques (parts made individually) and assembled with face-to-face mates.

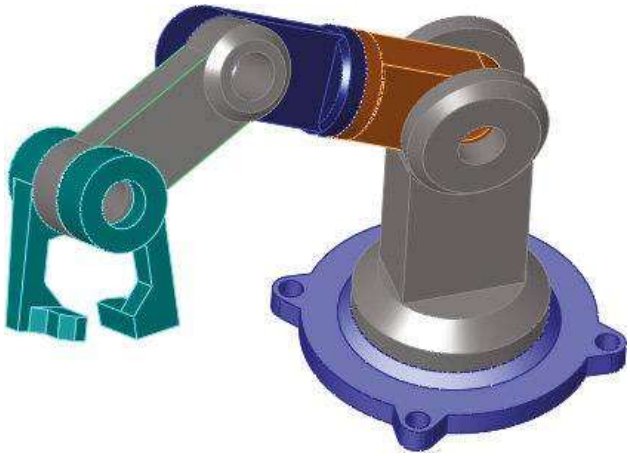


Figure 13.1
Mechanical parts using mates to locate and enable motion

But if you were to create, say, a scale model of a car, as shown in Figure 13.2, the method of independently designing each component, especially something like body panels, and then mating them together wouldn't make much sense. Methods in the other types of assemblies shown later in this chapter will help with this type of design.

Figure 13.2
Considering how you would design the parts of a scale model car

Using the "bottom-up with mates" method to design the body panels of the car so that they fit together well and looked smooth next to one another would be difficult or impossible.

Driving an Assembly with Sketches and Planes (Bottom-Up with Skeleton)

One way to avoid the potential pitfalls of mating to a base part is to replace the changeable faces and edges with items that are more stable. The stability hierarchy listing items from the most to the least stable looks like this:

- Assembly or part origin
- Assembly or part standard planes
- Reference geometry (plane, axis, point)
- Reference geometry from inserted parts (from using the Insert ➢ Part command)
- Sketch lines and midpoints
- Sketch endpoints
- Surface model faces
- Solid model faces
- Edges and vertex points
- In-context items
  - Reference geometry
  - Faces
  - Edges

An easier way to remember this without memorizing the list is that the more parents something has, the less reliable it is as a reference. This becomes more applicable if external references are involved, such as inserted parts or an in-context situation. Edges created by fillets or chamfers are lower on the list of stable references than other edges.

There is no clear answer to the question, "Is this reference stable enough?" It is entirely possible for you to be completely successful using in-context edges for all your model references. In order for that to happen, you have to plan your model very well and avoid any big topological changes (changes to the number or function of faces) to the model.

When you are building a part, selecting references from near the top of the previous list can be challenging, especially when faces and edges are so easy to use. You need to evaluate how much editing and rework you think you will generate when you must make changes for which you haven't necessarily planned.

Now consider the two examples mentioned in the previous assembly modeling method—the robot arm and the model car. You could design the robot easily with the sketch layout, but simulating the motion in 3D would be difficult if you did it in conjunction with the sketch. The model car would still be difficult to assemble, and if you were simply using sketches as the references between parts, it would be difficult to design the body panels such that they fit together smoothly.

Modeling Parts in Place (In-Context Design)

In-context design is discussed in more detail in Chapter 20, "Modeling in Context," but here you will get some idea of what to expect. Modeling parts in the context of an assembly that contains other parts enables you to make relationships between the parts. Those relationships are managed by the assembly. The parts have to be arranged spatially with respect to one another, and the references to the files must also be managed.

When you see a sales demonstration, the technique of using edges of other parts from the assembly to make a new part looks very compelling, especially when you make a change to the other part and the new part updates as well. It's hard to argue with that kind of functionality. But the price you pay for that sort of associativity is that you must manage the relationships between three files: the parent part, the child part, and the assembly. Furthermore, within the assembly, the relationships are made between specific instances of the parts, so if you have multiple instances of each, you must do something to remember which pair of parts is the driving pair.

Also, model history with parts doesn't work the same with assemblies. The relationship between the parts doesn't have any memory, so if you started the in-context relationships before the parent part was complete, and then put fillets over the edges that you had referenced, your in-context references would fail.

Take another look at the robot arm and the model car examples. Using in-context methods, you could certainly design the robot arm, but again you might run into some problems with getting it to move correctly while maintaining the references. However, with the model car, getting the parts in the right place wouldn't be any problem, because you would be modeling the parts in-place. On the other hand, you might be able to get the shape to flow smoothly, but it's still doubtful. In-context modeling can copy 3D surfaces between parts, but for an improved workflow for this type of work, you must read further into this chapter.

An example of a part where modeling in-context works well is a table with legs, as well as a fixture that sits on the table, as shown in Figure 13.3. The in-context work lines up the holes between the parts. There is no relative movement between the parts, and the individual parts aren't likely to be used in other assemblies.

The ideal situation in which to use in-context techniques is when two parts are assembled face-on-face, the shape of the contact faces are the same or offset, and holes are used for fasteners. The main requirement is that there is no relative motion between the two parts.

Modeling Parts as Multibodies

Another method you can use to model parts is to start the models in a multibody part. I don't recommend this method for creating finished parts as multibodies, but getting some of the major parts on an assembly started as single parts and then breaking them out into individual parts for details can be a very effective method.

Suppose you're modeling a riding lawn mower, and you need to create the plastic cowling on the front of the mower. The cowling is made up of multiple pieces because some of them are different colors and some are transparent. The complex shapes of the cowling encompass multiple parts. If you were to model one part and then try to model another part independently that shared some of the same shape, it would be very difficult or impossible to get the shapes to match acceptably.

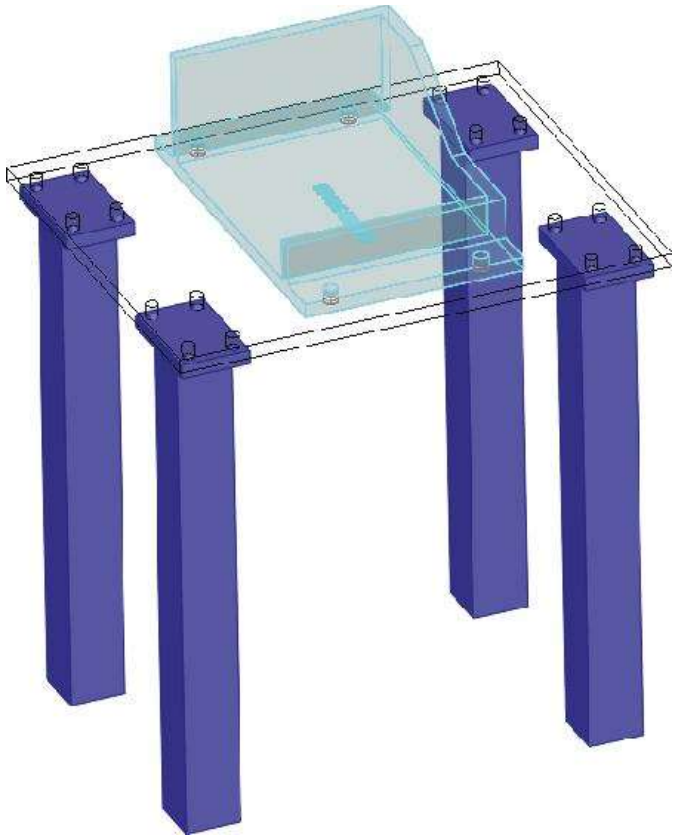


FIGURE 13.3
Using the in-context method to its best advantage

One answer to this problem is to create the shape in a single part, then break the single part into individual bodies, and then save the bodies external to the original part. When you put these parts back together into the assembly, each part can be placed so its origin matches with the assembly origin. Because all the parts started from the same part, they will share the same origin. This makes putting the parts back together much simpler. It makes assembly for motion more difficult, but parts that have a shape in common are more likely to be fixed with respect to one another.

Multibody modeling has advantages over in-context modeling in that it reduces external references (although saving bodies out as parts creates an external reference), but it also has some drawbacks. If you were to take all the features of individual parts and stack them into a single feature tree in a single part, you would probably be unhappy with the result. By making all the features for all parts within a single part file, you make troubleshooting much more difficult, and rebuild times are dramatically increased. Add to this the inability to reuse parts, do individual revision management, or perform simple assembly operations such as dynamic motion, exploded views, or BOMs, and following the multibody method through to finished parts becomes very unattractive.

The best option for using multibodies to create parts for an assembly is to start the parts in Multi-body mode, and then as soon as the interbody references are no longer needed, transition the bodies to separate parts.

Multibody modeling may not do so well when parts are repeated or where purchased components represent a large percentage of the total parts. Although you do have mate-like

functionality for placing bodies within a multibody part, it is probably not the best use of this method. Figure 13.4 shows a product that's designed as a multibody part, but it involves many difficulties because of reused parts and hardware.

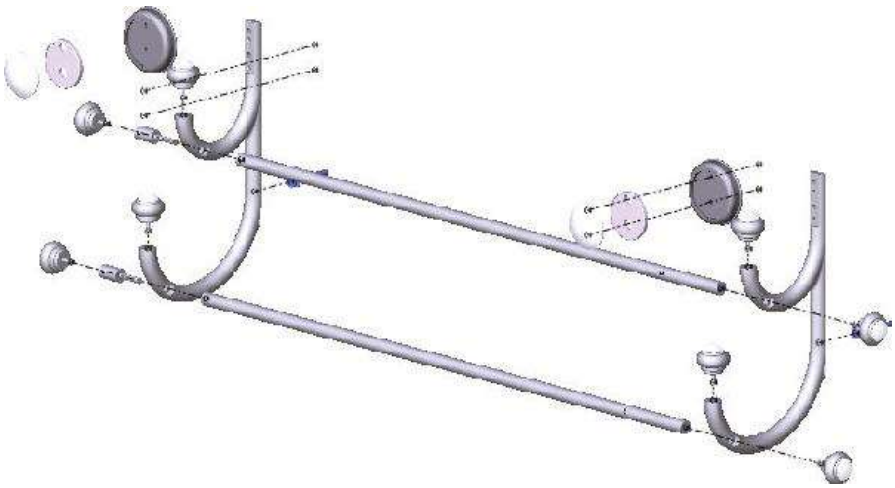


Figure 13.4
Reusing parts is not a strength of multibody methods.

Revisiting the test for each method, you would find that the robot arm is well suited to being designed as a multibody part and then reassembled with mates in an assembly. In fact, the multibody method is probably the best method for this type of work, because it maintains references between parts and then assembles the parts into an assembly mechanism with motion.

The model car, with its shape that flows between parts, would still be awkward, although it could be done as individual parts. Let's look at the last method to see if this helps with the car model.

Inserting a Master Model

You will learn about the master model technique in Chapter 33, "Employing Master Model Techniques." In a nutshell, a *master model* is a single part where you place sketches, reference geometry, surfaces, and maybe some solids, and then you insert that part into other parts to use a reference to build each individual part. Using this technique makes in-context work unnecessary, and it eliminates some of the dangers of creating too many features in a multibody part.

You assemble parts in this manner the same as with the multibody part method, and then you drop each part into the FeatureManager of the assembly. This aligns the part origin with the assembly origin, and because each part was built from the same master model, all parts share the same origin.

Take another look at the robot arm and model car examples. The robot arm may be a little awkward using this method, but it works. The multibody method is probably best for this type of design.

On the other hand, the master model method brings real power to projects such as the model car. You can design the entire outside of the car as if it were a single part and then break it into individual parts. In Figure 13.5, notice how some parts that will be manufactured as a single part can be easily pulled off the master model. Sketches within the master model can help define breaks between parts and can help determine if there is relative motion—for example, with the doors.

FIGURE 13.5
Pulling parts off of the master model

EXCLUDING SOME PARTS

Methods such as those mentioned previously are great for any part that is unique to an assembly, but should not be used for library-type parts. If you have a part that will be used in more than one assembly, it should not have any external references. All these methods create external references except *bottom-up assembly* (where parts are modeled individually and then assembled to one another or to a skeleton).

Any library parts that you have—and standard hardware such as nuts, bolts, washers, and so on—should be modeled without references. You should not create them as a part of multi-body parts.

Some SolidWorks users think that eliminating the distinction between assemblies and parts would be a good thing. That point of view works only for the simplest small assemblies with simple parts. It's easy to come up with situations in which the tree management tools required to maintain models built using that philosophy through changes do not even exist in the software.

Creating an Alternative to Multiple Assemblies

Re-creating or copying assemblies for different uses may seem very inefficient. In Chapter 19, "Controlling Assembly Configurations and Display States," you will learn about another way: using assembly configurations. Assembly configurations are a great tool with lots of theoretical benefits and practical limitations. As with most other functions in SolidWorks, when you need to create assemblies for multiple purposes, you may find that assembly configurations meet your needs. Or you may find it easier to just save a copy of the assembly, knowing that changes for the sake of rendering do not diminish the usefulness of the data for something like exploded views or managing external in-context references.

Identifying the Elements of an Assembly

As the number of parts and design requirements for an assembly grows, you may need to add some of the following types of assembly elements:

- Assembly equations
- Assembly Layout feature
- Assembly layout technique
- Assembly reference geometry (plane, axis, point, coordinate system)
- Parts
- Subassemblies
- Folders for parts
- Folders for mates
- Mates
- Assembly features (cuts that are made after the parts are assembled)
- Component patterns
- Mirror components
- In-context reference placeholders
- Smart Fasteners
- Smart Components
- Virtual components
- Envelopes
- Assembly configurations
- SpeedPaks
- Display states
- Assembly Design Tables
- Assembly Bill of Materials (BOM)
- Hidden/Suppressed/Lightweight/SpeedPak performance techniques
- Sensors
- Hole Series

You may already be familiar with some of these elements from having worked with part documents. They are shown in Figure 13. 6 and described in detail throughout this book.

FIGURE 13.6
Elements of an assembly

Working with Assembly Equations

Assembly equations work like part equations, but with some additional complications and considerations. For example, one of the additional features of assembly equations is the ability to drive the dimensions of one part from another part. The syntax is slightly different for assemblies, as shown in Figure 13.7.

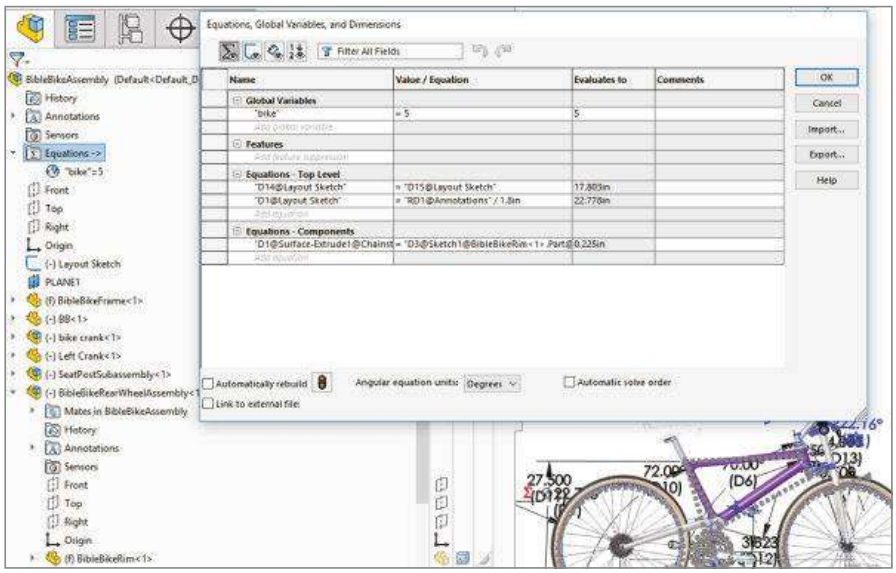


Figure 13.7
An assembly equation driving one part from another

Solving External References

Notice the arrow symbol (→) after the Equations folder in the assembly FeatureManager. This means that there is an external or in-context reference. An *external reference* means that an aspect of one part is dependent on something outside of the part. This has file management implications because you must maintain the names of the files so they always recognize the other file involved in the external relation. *In-context* means that one part has a relationship to another part in a position determined by an assembly. In this case, the in-context external reference can be solved only if the original part, the referenced part, and the assembly where the relationship was created are all open at the same time.

Understanding Global Variables

Global variables also work in assemblies, but they don't work between parts. Local assembly sketches can use these functions, and the parts can use them when edited in the context of the assembly.

Renaming

Equations update with new part names regardless of how you rename the parts. Names of subassemblies also update when you rename assembly files. This includes renaming a document using the Save As command, using SolidWorks Explorer, or using Windows Explorer. It also includes redirecting the assembly to the new part name, as well as renaming the assembly using each of these techniques. If the assembly can find the part and recognizes the part as the one it's looking for, then the equation will work.

Some of the methods mentioned previously for renaming parts are not recommended. SolidWorks Explorer and the Save As methods can be effective when used properly. References between files are a different issue from an equation's references to local filenames.

Recommendations

Although assembly equations are certainly a valid way to control part sizes, you should use assembly or part configurations, possibly with design tables, to accomplish something similar. Equations and configurations don't mix well, because the two methods conflict over which one controls the dimensions. Configurations with design tables are recommended over equations.

CAUTION You may have unexpected results if a single dimension is controlled from more than one location. For example, if you have a part-level equation and an assembly-level equation, then one of the equations will be automatically set to Read Only and will not be used.

SolidWorks Explorer

SolidWorks Explorer is a simple file-management tool that you can use with or without SolidWorks. You can access it through the menus at Tools ➢ SolidWorks Applications ➢ SolidWorks Explorer. When initially opened, it looks like a search function, but you can expand it into a Windows Explorer–type window with specific SolidWorks functionality such as adding tags to parts, properties, References, Where Used, Configurations, and more. (See Figure 13.8.)

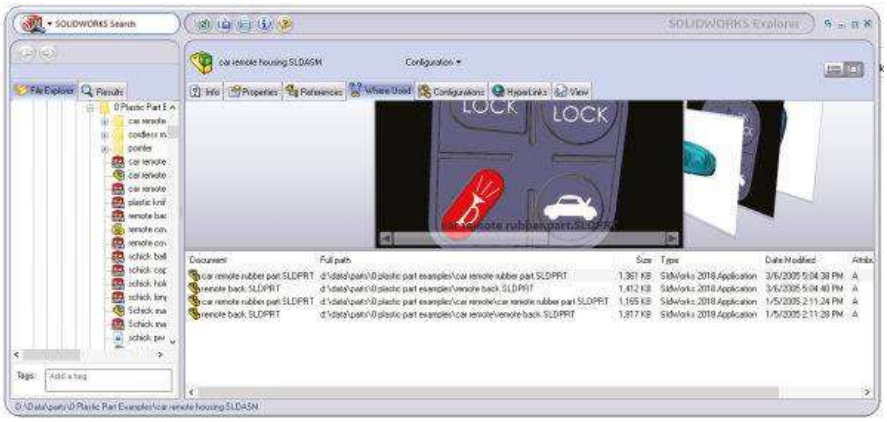


Figure 13.8 SolidWorks Explorer helps you manage several aspects of files and assembly references.

SolidWorks Treehouse

SolidWorks Treehouse enables you to create the framework for a product assembly using empty part, assembly, and drawing files. You can then use SolidWorks to build actual geometry within the files created by Treehouse. A lead engineer or manager would use Treehouse to set up the structure of the assembly—complete with subassemblies, parts, and drawings—and then plan the work to document the complete design. (See Figure 13.9.)

In Treehouse, you can:

- Create and organize assembly structure
- Create files using SolidWorks document templates
- Name files
- Create and edit properties

- Create and edit configurations
- Include existing documents from Windows Explorer

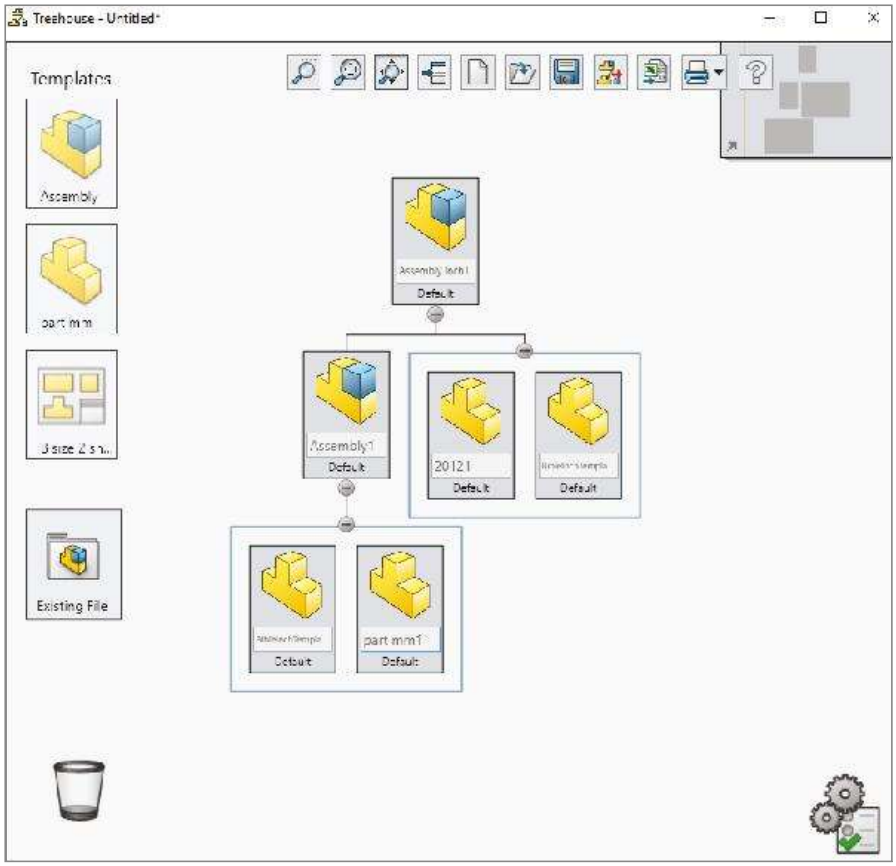


FIGURE 13.9
SolidWorks Treehouse helps you create a blank assembly structure.

To start working on a new assembly, put your cursor over the Assembly icon, and then drag an assembly template into the work area. To put a part into an assembly, put your cursor over the Part icon, and drag a part template onto the Assembly icon placed earlier.

Using an Assembly Layout Sketch

SolidWorks has an assembly feature called Layout that uses a 3D sketch to lay out the major functions of an assembly and even details of parts. The word *layout* also refers to a long-standing technique using 2D sketches in an assembly to do exactly the same thing. The distinction between the technique and the formal assembly feature is bound to be confusing. The Layout feature is relatively new and works only in assemblies, but layout techniques have been used in parts as well as assemblies for many years. This is simply a case of the unfortunate naming of a specific function with a name already used by a general technique. Let's just acknowledge this is confusing and move on.

When you look at the two functionalities, the feature is definitely intended to be used as an in-context tool, although you can more easily use the technique as a reference for controlling part position (through mating) rather than as a way to directly control the sizes and shapes of the parts. So when you see a reference to a *Layout* (capitalized), this refers to the formal feature. When you see a reference to a *layout* or *layout sketch* (lowercased), this refers to a technique

where a sketch is used at the part or assembly level to control geometry or part placement in some way.

The layout sketch is a very useful tool for laying out a mechanism in an assembly or even details on parts within the assembly. Sketches in the assembly have the same characteristics as they do in the part environment. In Figure 13.10, the assembly layout sketch is indicated with a heavy, dashed line for emphasis.

FIGURE 13.10
An assembly layout sketch controls the geometry of the frame and the overall bicycle assembly.

When combined with in-context techniques, assembly layout sketches can help to determine the shape of parts or the location, size, or shape of features within the parts. You also can use layout sketches to mate assembly components to far more robust and dependable mates, rather than mating part to part. The sketch shown in Figure 13.10 is used for both of these techniques. The shape of the frame and the major pivot points are established in the 2D sketch. The wheels also are mated to the sketch.

When you use an assembly layout sketch for either the in-context part building or simply part positioning, its main advantage is that it gives you a single driving sketch that enables you to change the size, shape, and position of the parts. You can use as many layout sketches as you want, and you can make them on different sketch planes. This enables you to control parts in all directions.

CAUTION When using layout sketches, it is assumed that the relationships are created such that the sketch drives the parts. However, nothing prevents you from using other elements in the assembly to drive the sketch. You should avoid this type of conflict, called a *circular reference*. It can create sketches that change with every rebuild and can seriously impact rebuild times. When using any type of *in-context relations* (relationships between items in an assembly), you need to be careful to establish one or more driving entities, which are not in turn driven by other entities.

To take this a step further, it is best to avoid *daisy chaining*, where A drives B, B drives C, and so on. A better practice is to make A drive both B and C directly. This saves on rebuild times and trouble-shooting and reduces future problems with lost references.

One of the drawbacks of this technique is that you give up dynamic assembly motion. To move the parts, you have to move the sketch and rebuild. The part does not move until the sketch is updated. If you need to combine layout functionality with dynamic assembly motion, refer to the discussion of the Layout feature in Chapter 16, "Working with Assembly Sketches and Layouts."

Working with Virtual Components

Virtual components are parts that are saved so they are internal to the assembly. You can save them out so they are external to the assembly and can be reused in other assemblies. You also can convert external components to virtual components. *Virtual components*, as the name suggests, can be either parts or subassemblies.

You may consider using virtual components for certain types of parts that need to be in the assembly but might not require drawings, such as glue, paint, oil, and so on. You may even use virtual components to model purchased subassemblies that flex; for example, you could create a hinge, and use the hinge assembly as a part in a library. However, if you choose to use virtual components, make sure that they won't cause you any difficulties downstream in file management, data sharing, or other requirements.

Creating Assembly Reference Geometry

Planes and axes are frequently created within assemblies to drive symmetry or placement of parts. You can use assembly layout sketches to create the reference geometry entities. When you create reference geometry within the assembly in this way, be aware that the normal parent/child relationships are still followed. The familiar icons for reference geometry entities are also used in the assembly tree.

Comparing History-Based and Non-History-Based Portions of the Assembly Tree

Because features such as sketches and reference geometry are history-based and found in the assembly tree, at least a portion of the assembly FeatureManager is history-based. However, not all of it is. For example, the list of parts and subassemblies is not history-based—the order does not matter at all to the software.

Sketches and reference geometry may appear before or after the list of parts, subassemblies, and mates. All the remaining entity types that can be found in the assembly FeatureManager are also history-based features, and you can reorder them in the tree. However, several situations can disrupt the process. Under normal circumstances, sketches and reference geometry at the top of the assembly FeatureManager are solved, then the parts are rebuilt if required, and then the mates are rebuilt. This ensures that the sketches and reference geometry are in the correct locations so that if parts are mated to them, all the components end up being the correct size and in the right position.

Assembly-level reference geometry can be created that references component geometry instead of layout sketches. This creates a dependency that changes the usual order. For example, the planes are usually solved before the part locations, but when the plane is *dependent* on the part location, the plane must be solved *after* the part. If a part is then mated to the plane, you will begin to create a dependency loop, such that the plane is solved, followed by the part, then the plane again because the part has moved, and then the mate that goes to the plane must resolve the part.

Best Practice

If you are a bit confused by all this, don't worry. You can simply follow this rule: Do not mate to anything that comes after the mates in the assembly FeatureManager tree. This includes assembly planes or sketches that are dependent on part geometry, as well as assembly features such as cuts, in-context features, component pattern instances, Hole Series, or Smart Fasteners.

This is probably a lot of information to absorb if you are a new user, but if you remember this rule, you can avoid creating models with circular references, where A is dependent on B, which is dependent on A—a never-ending loop that causes major problems for large assembly rebuild times.

Understanding Parts and Subassemblies

Parts and subassemblies are shown with their familiar icons in the design tree. You can reorder and group them in folders (covered in the next section) and edit the hierarchy of parts and subassemblies within an upper-level assembly.

The primary task of parts and assemblies is to help you organize your data. Information that relates to the geometry of a single manufacturable item is put into a part. Information that relates to the relationships between the parts is put into an assembly.

You may hear some people argue that parts and assemblies don't need to have different file types—that a part file should be able to handle both the geometric data and the relations between items. Organizing the data into different file types is necessary because it helps your computer know when to calculate which data. For example, if your computer had to rebuild all the features in every part as well as all the mates, rebuild times would suffer greatly.

In addition, parts in an assembly are different from bodies within a part. If you have bodies within a part, all your features are in one big list, rather than segmented into individual lists for each part. This is important for three reasons: rebuild times, troubleshooting errors, and reusing data. Part features that are in a big list with other part features cannot be organized or separated easily for other purposes.

Organizing Mates

The Mates area remains a constant, single folder, but you can organize it by reordering the mates and grouping them into folders. Each mate is shown with a symbol corresponding to the type of mate it is, but the Mates folder is shown as a pair of paper clips.

TIP In previous versions, the Mates folder was broken up into various mate groups when the model had a mate that caused a history-imposed rebuild. Therefore, mates in each group were rebuilt as folders. This had the effect of confusing a lot of people, so SolidWorks combined them all into one. Combining all the mates into a single folder allows you to ignore when you make a mistake and mate to a history-based item in the tree, thereby causing another mate group to be created. This is an example of SolidWorks trading ease-of-use for lack of control.

Mates are typically shown at the bottom of the assembly FeatureManager in their own area, but they are also organized in a folder underneath each individual part, so you can see the mates that each part is involved in, as shown in Figure 13.11.

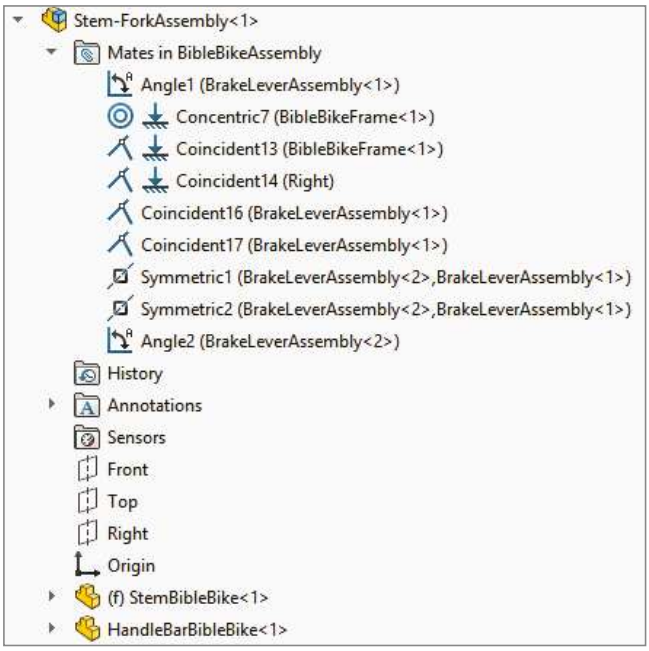


FIGURE 13.11
Mates organized in a folder under parts in an assembly

The symbols that look like ground next to the mate symbols help you identify the mates that connect the part to ground, or another fixed part that prevents the mated part from moving.

Another way of organizing mates is to change the way the assembly FeatureManager is displayed. There is a pair of settings that work together. One is the Show Hierarchy/Feature Detail toggle, and the other is the View Features/Mates And Dependencies toggle. If you choose the Hierarchy and the Mates And Dependencies options, you can get a tree where the mates are shown under the part or assembly names, as shown in Figure 13.12.

The Show Hierarchy/Feature Detail toggle is found in the top-level assembly RMB menu. The View Features/Mates And Dependencies toggle is found in the same RMB menu, under Tree Display, which is shown in Figure 13.13.

Applying Assembly Features

In manufacturing, after parts are assembled, secondary machining operations are sometimes applied to them to ensure that holes line up properly, or for other purposes. For example, assembly features can be cut extrudes, cut revolves, or hole features. These features appear only in the parts at the assembly level, not in the features of the individual parts.

You should not confuse assembly features with in-context features. *In-context features* are created when you are editing a part in the assembly with a reference between parts, but the sketch and feature definition are in the actual part.

Assembly features still cause additional assembly complexity, and if you mate to an assembly feature, you cause another hitch in the assembly rebuild cycle (and formerly would have created another mate group). If you need this kind of feature, that's fine; make it, but be extra careful not to mate to it or create in-context references to it.

If functionality in SolidWorks seems powerful, it probably is. But there is always a price to be paid for power, and the cost is usually in rebuild speed or more complex troubleshooting if you have problems.

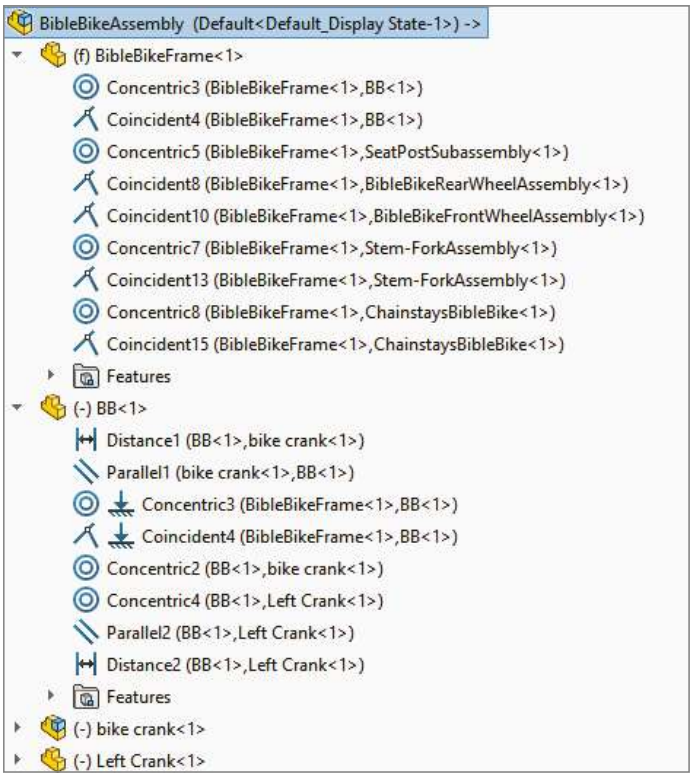


FIGURE 13.12
Showing Hierarchy and Mates And Dependencies

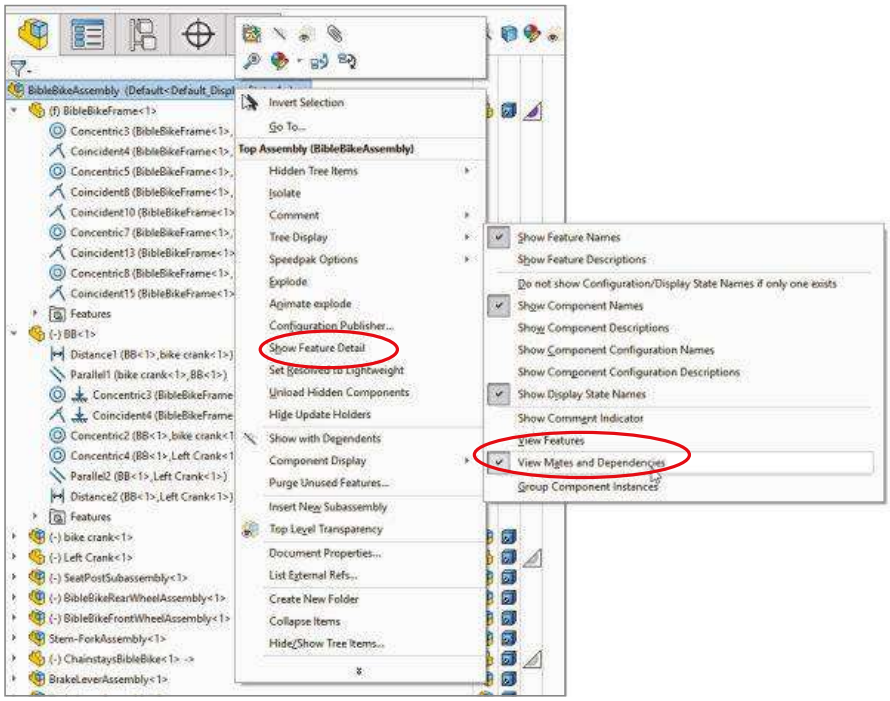


FIGURE 13.13
Settings for organizing the display of mates

Using Component Patterns and Mirror Components

Component patterns can pattern either parts or subassemblies by creating either a pattern defined in the assembly or a pattern that follows a pattern feature created in a part. The pattern is listed as a feature in the assembly FeatureManager, and all the instance parts appear indented from the pattern feature in the design tree. You can hide or suppress each instance, change its configuration, and in most ways control it as if it were a regular part in the design tree.

Because the options for locally defined patterns are comparatively limited, users generally like to use part feature patterns to drive the component patterns when possible.

Component patterns are listed at the bottom of the assembly FeatureManager with a set of components under a LocalPattern icon. The component instances under the LocalPattern can be controlled in several ways, including through assigned configurations, colors, and display states. The pattern can even be dissolved, leaving the components but removing the intelligent pattern that places them.

Mirror components are listed under a special MirrorComponent icon after the mates.

PERFORMANCE

To improve performance, you should pattern subassemblies if possible. If it's not possible, patterning a group of parts is the next best option. Making multiple patterns, one for each part, is an inefficient way to accomplish the same thing.

Using SpeedPaks

A SpeedPak is a derived configuration of an assembly that keeps only selected solid bodies and faces but can represent the rest of the assembly with nonselectable display data. You can use a SpeedPak to represent an entire subassembly within an upper-level assembly. SpeedPaks are intended to increase performance with very large assemblies and drawings.

Figure 13.14 shows the SpeedPak PropertyManager on the left, which you access by right-clicking an active configuration and selecting Add SpeedPak. Each configuration can have only one SpeedPak.

The center image in Figure 13.14 shows the configuration list with the SpeedPak indented under the Default config, and the entire assembly. The right image shows the SpeedPak inserted into an assembly document, consisting of a single face and two solid bodies. Notice the special icon associated with SpeedPaks. You can change a part in an assembly from or to a SpeedPak in the same way that you would change a configuration using Component Properties.

Remember that this is a tool for increasing assembly speed, and you must always give up something to increase speed. A SpeedPak is similar to Lightweight assemblies and components in that it's display-only data. If your expectations of the tool match its actual functionality, you will be very satisfied with what the SpeedPak offers. For this reason, it's important to understand the abilities and limitations of SpeedPaks.

Using Ghosts

You can use any faces or bodies that you select in the Include lists either manually or through the Quick Include sliders (which automatically select bodies and faces based on size) in assemblies to mate to or in drawings to dimension to. Any geometry that isn't selected is included as a ghost: It displays, but you cannot select it. When you move the cursor near ghost geometry, the ghost fades away, revealing only selectable geometry. Notice at the bottom of the SpeedPak

PropertyManager that you can also choose to remove the ghost data and further increase the memory savings.

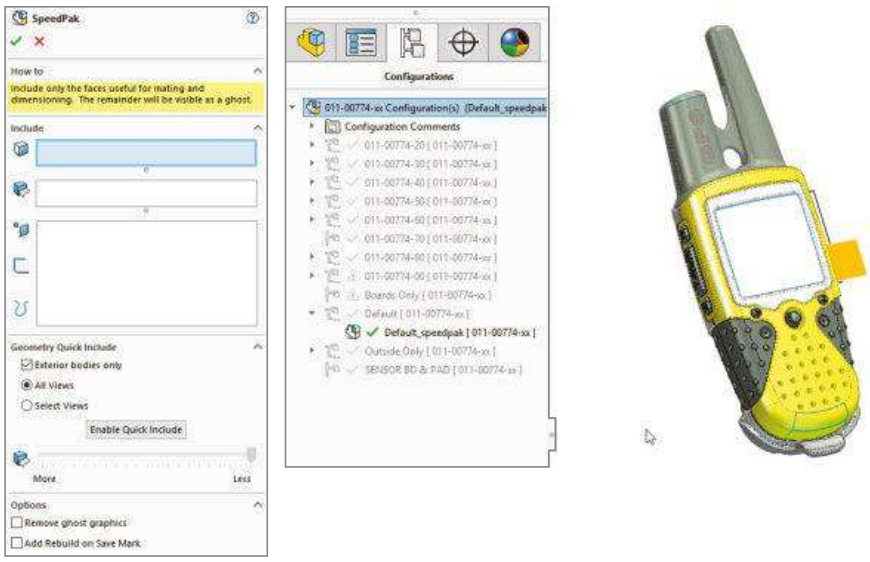


FIGURE 13.14
Managing SpeedPaks
Model of Garmin assembly from the SolidWorks demo sets

Sharing Self-Contained Data

The SpeedPak is self-contained. All the selected face and body geometry is saved inside the assembly. If you want to send someone a visual representation of an assembly, you can make a SpeedPak configuration and send only the assembly file; no parts are required. This is the equivalent of being able to put an eDrawing file into an assembly.

Using SpeedPaks with Drawings

You can even use SpeedPaks with drawings. Just remember that only edges created by the faces or bodies in the Include lists can be dimensioned to. Some functionality exists for the ghost data, such as BOM inclusion and numbered balloons. Ghost data displays as gray on the drawing, while geometry in the Include list is black.

Using Subassemblies

The primary tool for organizing assemblies is the subassembly. A *subassembly* is just a regular assembly that is used as a component in another assembly.

BEST PRACTICE

You are not limited to a specific number of levels of subassemblies, although for different sizes and types of assemblies, you should establish a best practice for your company. For example, you might establish a guideline that suggests that assemblies of 100 parts or fewer go no deeper than three levels.

You can use several criteria to determine how subassemblies are assigned:

- Performance
- BOM
- Relative motion
- Prefabricated, off-the-shelf considerations
- According to assembly steps for a process drawing
- To simplify patterning

The underlying question here is based on the multiple functions of your SolidWorks assembly model. Is the assembly intended primarily for design? For visualization? For documentation? For process documentation? When used primarily for design, the assembly is used to determine fits, tolerances, mechanisms, and many other things. As a visualization tool, it simply has to look good and possibly move properly if that's part of the design. As a documentation tool, how the model relates to the BOM is important, and so is the order in which subassemblies are added. When you are using a subassembly as a process tool, you need to be able to show the assembly in various intermediate states of being assembled, likely with configurations.

Creating Subassemblies from Existing Parts

You can create subassemblies from parts that already exist in an assembly. To do this, select the parts you want to add to the subassembly by pressing and holding the Shift or Ctrl key, or using box selection techniques, and then selecting Form New Subassembly Here from the right mouse button (RMB) menu. You will prompted to assign a name or possibly select a template for the new subassembly.

CAUTION When you're creating a new subassembly from existing parts or moving parts into or out of a subassembly from the upper-level assembly, some things may be lost. For example, mates are moved from the upper level to the subassembly. If you have in-context relationships, they may be removed. You cannot easily undo operations that create subassemblies.

After you have created the subassembly, you can add or remove components using the drag-and-drop method. For example, Figure 13.15 shows two different actions with two different cursors. On the left is the cursor that indicates that the part named BB is being moved into the subassembly named bike crank. On the right is the cursor that indicates BB is being reordered after the bike crank. To move a part out of a subassembly, you can simply drag the part into the upper-level assembly.

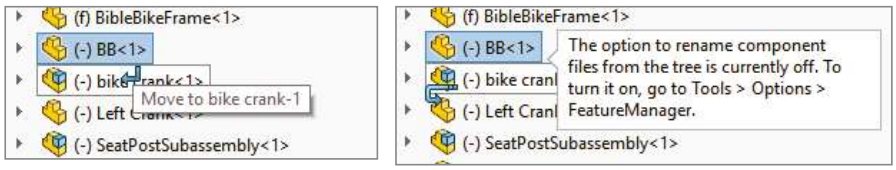


FIGURE 13.15 Moving parts into a subassembly

NOTE When you are dragging a part out of an assembly and into another one, you may again see the cursor symbol that appears in Figure 13.10. If you do not want this to happen, press and hold the Alt key while dragging. The cursor symbol will change to the Reorder cursor (a reversed, L-shaped arrow), and the part will be placed after the subassembly rather than within it.

Inserting a New Subassembly

Along with the RMB menu option Form New Subassembly Here, which takes existing parts and puts them into a newly created subassembly, you can use another option called Insert New Subassembly. The names of these functions don't adequately describe their different functions. Insert New Subassembly inserts a blank subassembly at the point in the design tree that you indicate by right-clicking it. You can place components into the subassembly by dragging and dropping them from the main assembly, or you can open the assembly in its own window and insert parts by using the usual methods, such as drag-and-drop, or the Insert ➢ Component tool.

Dissolving Subassemblies

If you want to get rid of a subassembly but want to keep its parts, you can use the Dissolve Subassembly option through the RMB menu. This option has some of the same consequences of the Form New Subassembly Here option in that mates are moved from the subassembly to the upper-level assembly, and you may lose in-context relations and assembly features.

Organizing for Performance

In SolidWorks, performance refers to speed. Subassemblies can contribute to speed-saving modeling techniques by segmenting the work that the software needs to do at any one time.

Solving Mates

The mates that contribute to putting the pieces of an assembly together are solved at the top assembly level. Under normal circumstances, subassemblies are treated as static selections of parts that are welded together, and their mates are not solved at the same time the top-level assemblies' mates are solved. This segmenting of the mates leads to improved performance by solving only one set of mates at a time.

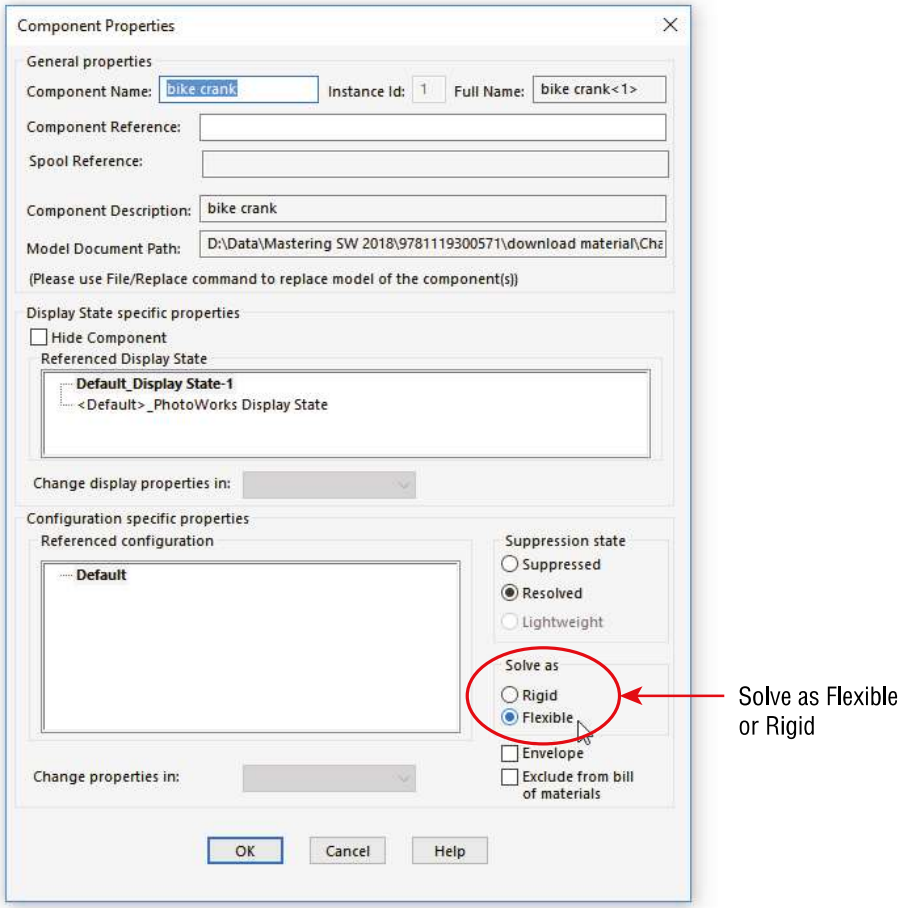
Mates are usually solved as a single group unless there is a special situation, such as mates to in-context features, component pattern instances, or an assembly feature, all of which have already been described in this chapter. When one of these situations occurs, the mates must be divided into separate groups or solved multiple times. This is done behind the scenes so the user doesn't have to worry about it. Multiple rebuilds affect the user only in terms of rebuild times.

Using Flexible Subassemblies

When you put a subassembly into an upper-level assembly, the mates for the parts of the subassembly are not solved in the upper-level assembly. This means that if a subassembly is a mechanism, the mechanism does not allow dynamic assembly motion in the upper-level assembly, and it is considered rigid. For example, in Figure 13.16, the front fork is a linkage mechanism, but it's also a subassembly. Without reassembling the parts of the fork in the upper-level assembly, you can allow the mates from the fork subassembly to be solved in

the upper-level assembly by using the Solve As option in the Component Properties dialog box, shown in Figure 13.16. When you select the Flexible option, you enable the mates of this subassembly to be solved in the upper-level assembly, which allows the parts of the subassembly to move in the upper-level assembly. To access the Component Properties dialog box, right-click the subassembly and select Component Properties from the menu.

FIGURE 13.16
Creating a flexible subassembly



Flexible subassemblies have become more reliable and easier to use. You should work with them or do some experimentation to see if they assist your modeling process. If you find they cause trouble in some situations, they are easy enough to deactivate.

Organizing for the Bill of Materials

The Bill of Materials (BOM) is a table that is placed either into a drawing of an assembly or in an actual assembly. This table shows the parts used in the assembly and includes other information, such as part numbers, quantities, descriptions, and custom property data.

Businesses often represent assemblies and subassemblies in various ways by using Manufacturing Resource Planning (MRP) or Enterprise Resource Planning (ERP) software. The methods that accountants and manufacturers use to organize assemblies are not always the same

as those that engineers or designers might choose, but some companies require the BOM on the drawing to match the MRP or ERP Bill of Materials.

Best Practice

When you are forced to model something in an unnatural way to satisfy an outside demand such as special BOM requirements, it might be best to detach the unnatural part and model normally. In the situation mentioned here—where MRP is forcing how the assembly is put together by requiring the BOM to match MRP—you should separate the BOM from the assembly structure rather than build an assembly that makes other SolidWorks functions difficult. This ensures that the BOM becomes a manually maintained document. Alternatives to this approach would be to make configurations or entirely new assembly documents to drive the BOM.

Grouping Subassemblies by Relative Motion

A more natural way to group subassemblies is by considering relative motion. In the bicycle example, each wheel is a separate subassembly because it moves as a unit relative to the rest of the assembly.

Grouping subassemblies by relative motion is great for assembly modeling, but it doesn't usually reflect product reality very well. Using this method, you often end up with parts in the subassembly that must be disassembled in order to actually put the physical parts together. However, if your only consideration is ease of modeling, then you should probably use this method.

Organizing Groups of Purchased Components

If you are modeling a product that is created from a shopping list of purchased components, then it may make the most sense to organize your subassemblies into groups of parts that are purchased together. In fact, purchased subassemblies are often modeled as single parts, except when relative motion is required in the purchased assembly.

For example, in the bicycle assembly, the sprockets on the rear wheel are purchased as a separate unit, but the part that mounts onto the wheel moves relative to the sprockets that are driven by the chain. This is an example of a purchased part that would be modeled as a subassembly to show relative motion. The bicycle chain, another purchased subassembly, has not yet been added to this assembly and is a more complex model. The desire to show all the individual links moving through the path may override both the complexity of assembling it and the performance considerations of exercising all the mates.

Although the BOM method of organizing assemblies sometimes leads to unnatural solutions, you should not discard it altogether. If you can devise concessions in order to make the BOM work automatically, then you should do this.

Depicting an Assembly Process

Manufacturing and assembly processes need to be documented as well as individual part design. You often need to create exploded-view assembly instructions for manufacturing or service documentation at each step of a multistep assembly process. Figure 13.17 shows an example of this type of process documentation.

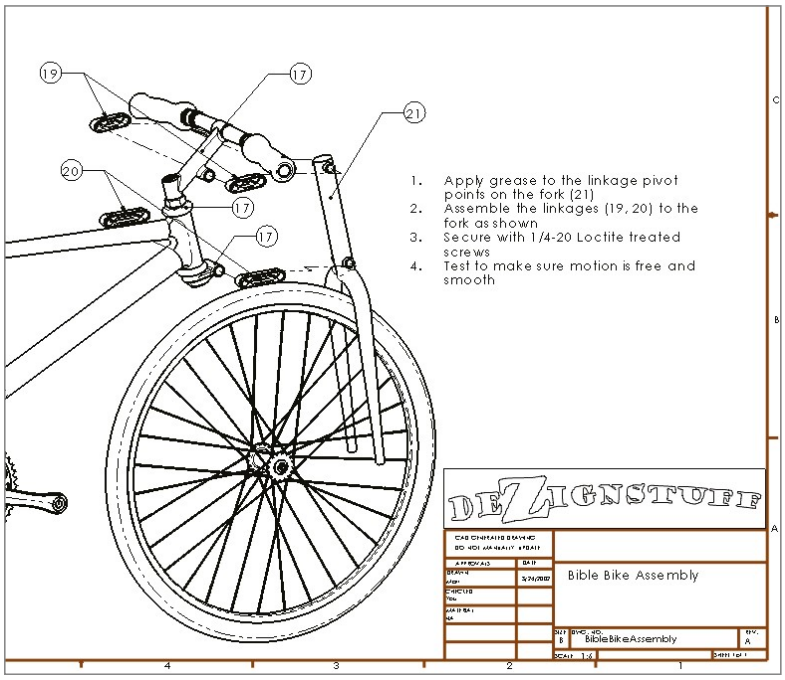


FIGURE 13.17
Assembly process documentation

This task is certainly different from the initial design or modeling of the assembly, and it may require an entirely separate assembly model. Generally, you can perform the different steps by using a separate configuration for each process step, with exploded views for each configuration.

INFLUENCING ITEM NUMBERING

Balloons number the parts according to the item number used in the BOM, but of course, you don't know the item numbers until the BOM is created. You can influence the item numbers by reordering the parts in the assembly (which is discussed later in this chapter), by manually editing item numbers, or by manually numbering the balloons.

SEPARATING STEPS

Each step corresponds to an assembly configuration (discussed in Chapter 19, "Controlling Assembly Configurations and Display States"), and you can place them on a separate sheet of the drawing (discussed in Chapter 30, "Creating Assembly Drawings"). Each configuration can have multiple exploded views, if necessary, to show all the steps.

Patterning Considerations

The most efficient way to pattern large numbers of components in an assembly is to pattern a single subassembly with all the components to be patterned in it. Although this may not be easily combined with some of the other considerations mentioned previously, it's another option that you can use to organize assemblies.

Using Folders

Folders are primarily used in the assembly FeatureManager for grouping parts and mates into special classifications for easy browsing, or groups that you can easily hide and show, or suppress and unsuppress, as appropriate. Figure 13.18 shows some examples of these folders.

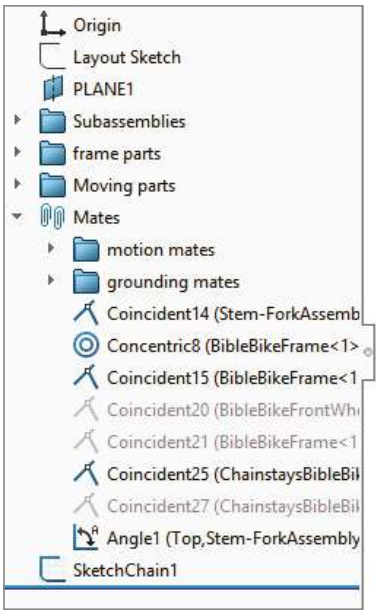


FIGURE 13.18
Folders that are used to organize components and mates

Creating Folders in the FeatureManager

You can add folders to the assembly FeatureManager in two ways:

- By adding existing components to a new folder
- By creating an empty new folder

USING ADD TO NEW FOLDER

To use the Add To New Folder tool, right-click a component or mate (or selection of components or mates) and select Add To New Folder from the menu. This moves the component or mate into the folder. Folders don't affect the assembly in any functional way; they are simply for organization, to speed browsing and selection.

USING CREATE NEW FOLDER

To simply create a new folder without putting anything into it right away, right-click either the Mates area or the Components list and select Create New Folder from the menu.

Reordering Items in the Tree

Sometimes, you may want to reorder items in the assembly tree. For example, you may want to place items close to one another in the tree, or you may be preparing to put items that are next to one another into a single folder. You also may want to reorganize components for the BOM display.

You can reorder mates simply by dragging them. Mates display in the order in which they are created, but the order is not significant. You can reorder them however you like.

Components also display in the order in which they are added to the assembly, and you can reorder them in any way you like.

Best Practice

It is often useful to have an ordering strategy that helps you work with the model. For example, you should try to keep the biggest parts, the parts that everything else is mated to, or the part that is treated as "ground" as the first part(s) in the assembly. Then put the fasteners and other cosmetic or BOM-driving parts at the end of the tree, usually in a descriptively named folder.

Working with Tree Display Options

Display options for items in the FeatureManager are often overlooked but can be useful for displaying data about parts, subassemblies, mates, and features. Figure 13.13 shows the RMB options. You must right-click the top-level assembly name in the FeatureManager, and click Tree Display to access this menu.

NOTE All these options are available for parts and drawings as well, except for the View Features option and the View Mates And Dependencies option, which are related to assemblies.

Showing Feature Names and Descriptions

If you are so thorough that you have added descriptions to your features, then you are doing well. Figure 13.13 shows the options for displaying feature names and descriptions in the FeatureManager. Generally feature names are sufficient, but descriptions can add to search and filtering options in long feature trees.

Showing Component and Config Names and Descriptions

By default, SolidWorks uses the filename of a part or assembly as the component name. If you want to use some other name in the assembly FeatureManager, you must change a couple of settings.

First, go to Tools ➤ Options ➤ External References, and turn off Update Component Names When Documents Are Replaced. This enables you to keep the same component name even if you replace that component (you have to use the Replace Component functionality—as opposed to simply deleting and reinserting—in order for this to work).

Second, right-click on the component with the name you want to change and go to Component Properties, shown in Figure 13.19. Now you can change the Component Name (if you didn't do the first step, you will get a lengthy warning message). Notice the difference between the Component Name, which I changed manually, and the Model Document Path, which is filled in automatically. This dialog even has a handy hint on how to replace the component if you want to, with File ➤ Replace.

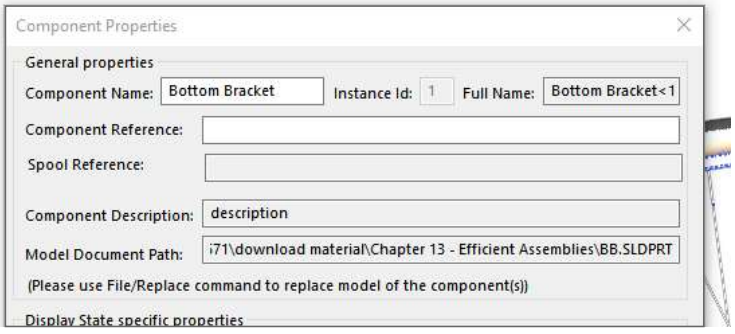


Figure 13.19
Changing a component name in Component Properties

Don't stop reading now; this gets slightly more complex. If you want to rename the component in the assembly FeatureManager without the Component Properties dialog, you would probably try the slow double-click or F2 Windows methods. But that wouldn't work unless you have selected. Go to Tools ➤ Options ➤ FeatureManager and turn on the setting "Allow component files to be renamed from FeatureManager tree." Then when you try to rename it from the FeatureManager, you get another SolidWorks warning as shown in Figure 13.20. At least here you can choose the Don't Show Again option.

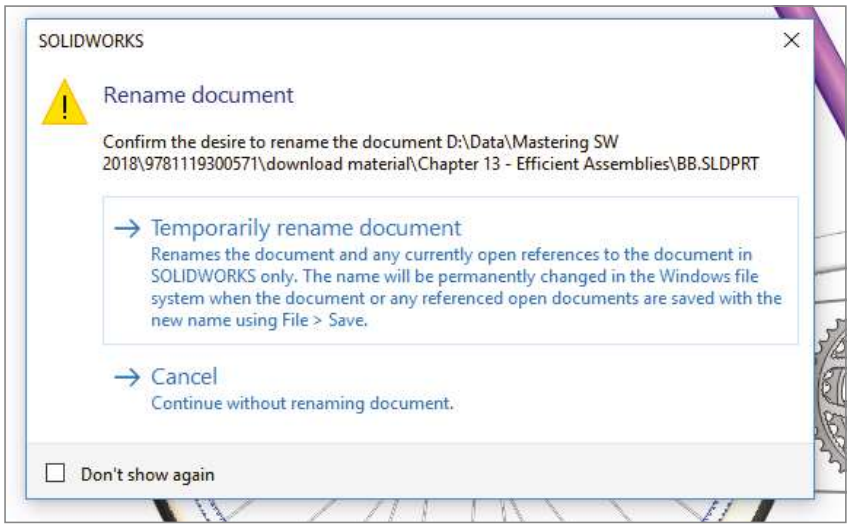


FIGURE 13.20
Renaming from the FeatureManager

Using Component Reference per Instance

In the Component Properties dialog box (refer to Figure 13.19), enables you to enter Component Reference information. This capability is typically used in electrical diagrams for similar components with different values, such as power ratings, capacitance, or resistance. Instances of the same component in a SolidWorks assembly that have the same Component Reference can be listed together on a BOM. Instances with different Component References are listed separately on the BOM. Figure 13.21 shows parts listed in a pattern with Component References listed for each instance.

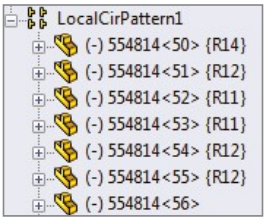


FIGURE 13.21
Listing parts in an assembly containing Component Reference information

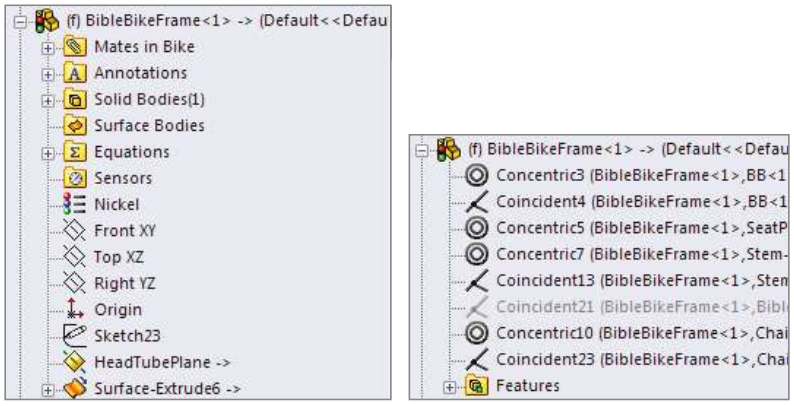
In order for Component References to be used in balloons on an assembly drawing, the drawing must have a BOM with a Component Reference column. BOMs are handled in detail in Chapter 31, "Modeling Multibodies," where the topic of Component References will be revisited.

Viewing Features, Mates, and Dependencies

The set of options shown in Figure 13.13 determines whether you see the part features or the assembly mates after the name of each component in the assembly tree. The default setting is for the part's features or the subassembly's components to display, just as if the part or subassembly were open in its own window.

The View Mates And Dependencies option can also show the features, but they are placed into a separate folder. This option makes it very easy to see the mates that are assigned to an individual component. For example, in Figure 13.22, the image to the right shows the mates directly under the BibleBikeFrame part. This often makes troubleshooting much easier because it isolates the mates for a single part. Notice also that the first folder under the part name in the image to the left in Figure 13.22 is the Mates folder. This indicates that, regardless of whether you choose to display mates or features, you always have easy access to the other type.

FIGURE 13.22
You can view features, as well as mates and dependencies.



View Features View Mates

The View Mates tool is extremely valuable for looking at how an assembly is held together with mates. When you right-click a component in the assembly and choose View Mates from the RMB menu, SolidWorks highlights the component you clicked and makes all parts that mate to that component transparent. Any parts that are not related are hidden. SolidWorks also displays a small dialog box with the list of mates touching the component you clicked.

Figure 13.23 shows this arrangement using the Bible Bike assembly. This is a big help in mate visualization.

If you Ctrl+select multiple components before starting the View Mates tool, SolidWorks no longer displays the common mates in bold format; it just lists them at the top of the dialog box.

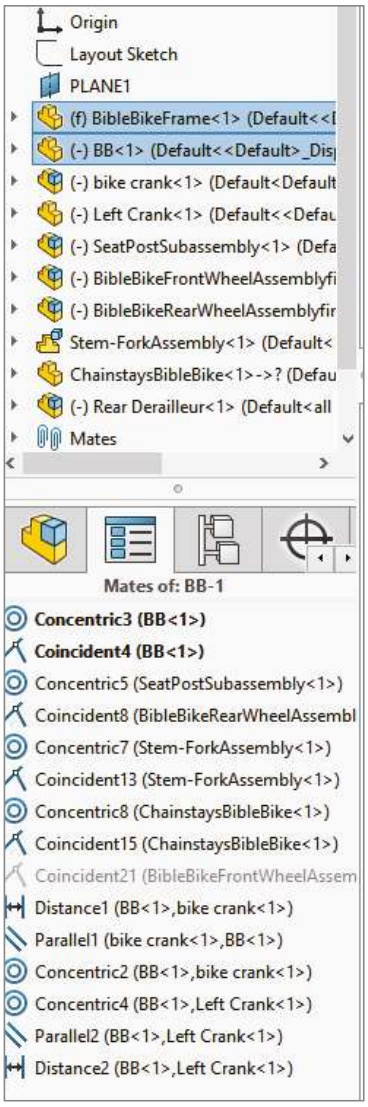


FIGURE 13.23
Showing mates in the PropertyManager pane

Tutorial: Arranging Assemblies

In this tutorial, you will take an assembly that is already put together, group its components into subassemblies, and then convert one subassembly into a flexible subassembly. Note that some of the commands and RMB options you are asked to select may not be shown on the truncated RMB menus. To remedy this, click the double-arrow at the bottom of the RMB menu or choose Tools ➤ Customize ➤ Options from the menu, and click the Show All button for both shortcuts and menu customization.

Follow these steps to learn how to effectively arrange items in an assembly:

1. Start by opening the `Robot Assembly .sldasm` file from the download material for this chapter.

   Notice that the filenames are long and somewhat difficult to read. This would also apply for files that use sequential numbers for the filename instead of a descriptive name.

2. To display a more readable name, right-click the name of the assembly at the top of the FeatureManager, select Tree Display from the menu, and then turn on Show Component Descriptions. Repeat these steps, and this time turn off Show Component Names.

   Figure 13.24 shows the display of the FeatureManager after the change. Even the top-level assembly uses its description rather than its filename.

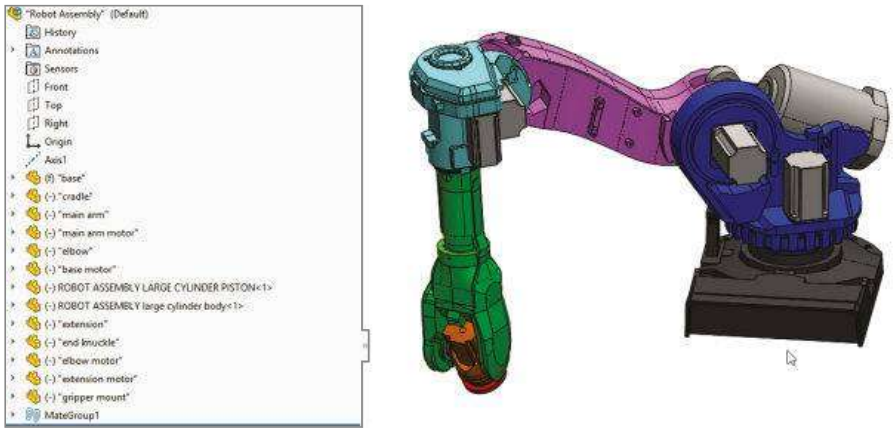


FIGURE 13.24
Simplifying the FeatureManager display to include descriptions

   Notice that two components still use their clumsy filenames rather than easy-to-read descriptions. This is because descriptions were never entered for those two components.

3. Open the Large Cylinder Piston part by clicking the part in the FeatureManager and then clicking the Open icon. Choose File ➢ Properties from the menu, and make sure the Custom tab is active. Create a new property called **description**, assign a type of text, and then enter **large cylinder piston** for the value. Save the part (Ctrl+S is a fast way to do that), and then flip back to the assembly (the fastest way to do that is to press and hold Ctrl and press Tab).

   If the display has not yet updated, press Ctrl+Q to force the tree to rebuild.

4. Open the Large Cylinder Body part. In this part, choose File ➢ Save As from the menu, and select Save As when prompted. Leave the name as is, but where it says Description, enter **large cylinder body**, as shown in Figure 13.25. Click Yes when asked if you want to replace the document of the same name. Flip back to the assembly when you are finished. You may have to rebuild to see the change update.

5. Press and hold Ctrl, and select the large cylinder piston and the large cylinder body parts from the FeatureManager. Then right-click and select Form New Assembly from the menu. If your assembly template has a description, it will appear in the FeatureManager. If not, the filename will appear.

   You have just created an assembly as a virtual component while the parts are external documents. There is no actual file corresponding to this entry in the assembly FeatureManager. If you switch the Tree Display to show filenames, you will see what is shown in Figure 13.26; the name of the assembly is `Assem1^Robot Assembly`. So the virtual component gets a default name (Assem1) followed by the name of the parent assembly (Robot Assembly) to ensure that it has a unique name, if there are other virtual components in other assemblies.

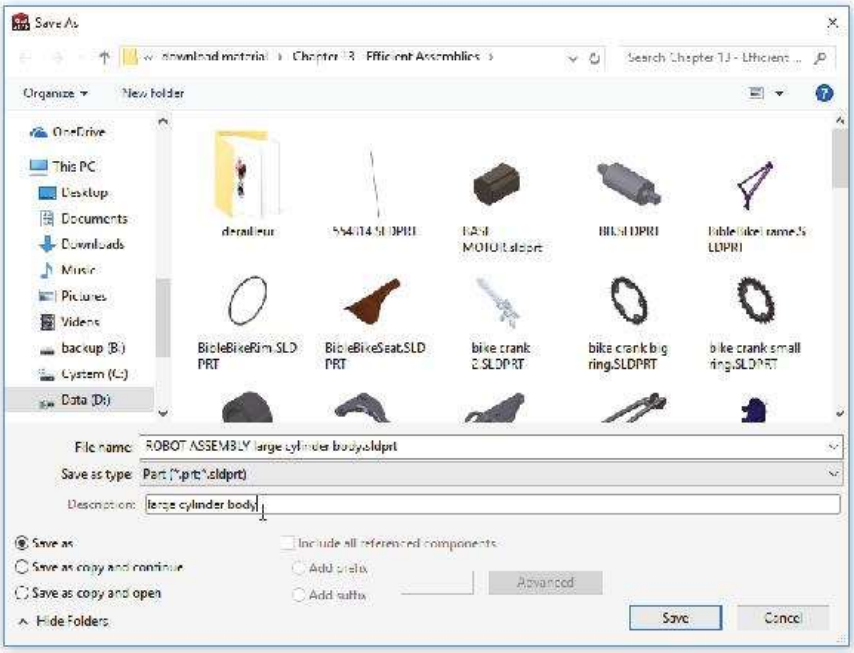


FIGURE 13.25
Adding a description in the Save As dialog box

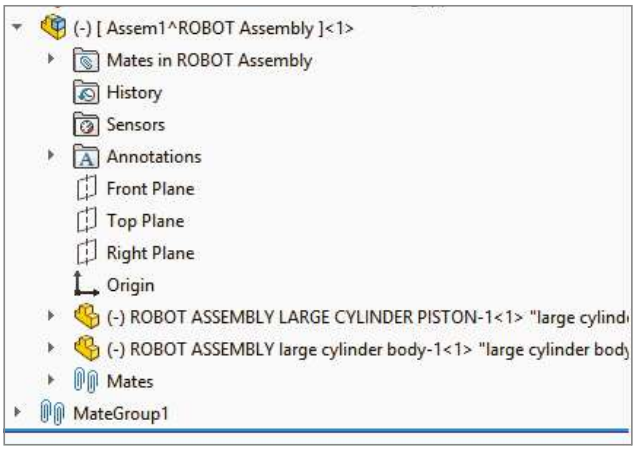


FIGURE 13.26
The newly created virtual component subassembly and its external parts

6. Press and hold Ctrl; select the Base Motor, the Main Arm Motor, and the Cradle; and make another new subassembly. After you create this subassembly, right-click it and select Save Assembly (In External File) from the menu (and select "Save only the subassembly to an external file"). Change the name of the assembly from Assem2 to **Cradle**. Save it in the same path as the rest of the parts.

   After a virtual component is saved externally, you cannot use the Undo command to reverse it, but you can right-click the external file in the FeatureManager and select Make Virtual from the menu.

   In Figure 13.27, notice that several of the mates came along into the new subassembly. These are the mates between the motors and the cradle. The mates that locate the cradle to the other parts in the assembly have remained in the upper-level assembly.

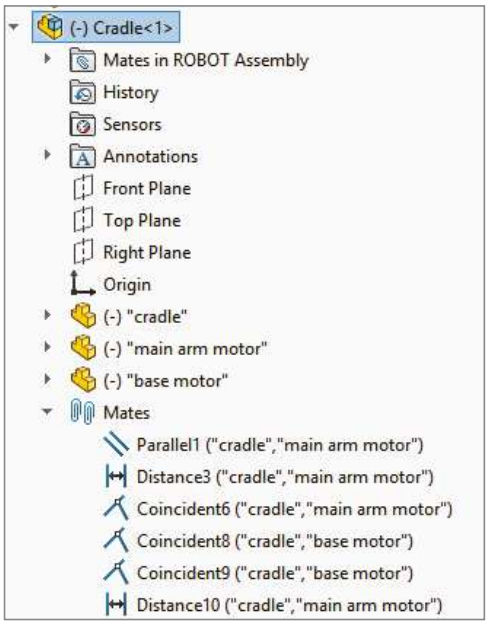


FIGURE 13.27
The cradle assembly brings along internal mates.

Note that you cannot undo the subassembly operations. If you need to remove the assembly but keep the parts, right-click the assembly and select Dissolve Subassembly from the menu.

If you try to move the parts in the assembly, you should notice that all parts work as they should, except that the Main Arm does not move up and down. This is because you turned the cylinder and piston into a subassembly, and the subassembly mates are not solved in the top level, which means the subassembly cannot move within itself.

To get around this, you need to make the subassembly into a flexible subassembly, which solves the subassembly's mates in the top-level assembly.

7. Right-click the cylinder subassembly in the FeatureManager, and select Component Properties from the menu. In the lower-right corner of the dialog box, there is the option to Solve As Rigid or Flexible. Change this setting to Flexible.

After you click OK and return to the assembly, the main arm will move as it did originally, and the piston will move in and out of the cylinder. Notice that the symbol for the assembly changes when it is changed to a flexible subassembly.

Tutorial: Managing the FeatureManager

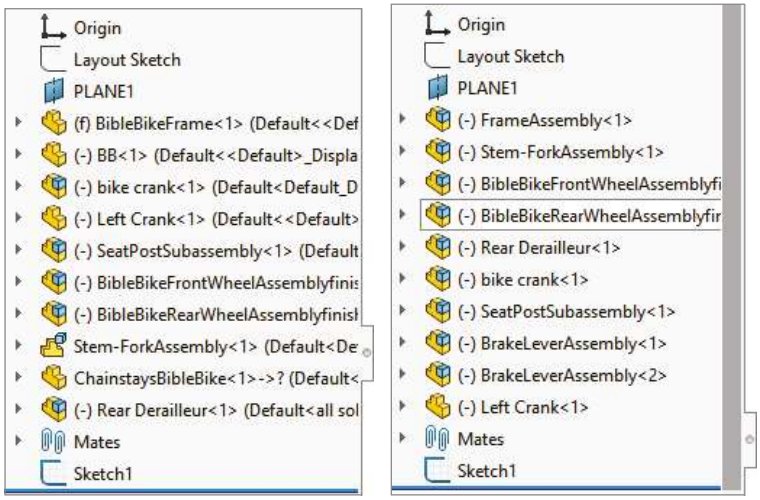
This tutorial uses the `BibleBikeAssembly.sldasm` file located in the Chapter 13 folder in the download material from the Wiley website. Open the file and follow these steps to learn about managing the FeatureManager:

1. Create a new subassembly within the existing assembly using the parts BibleBikeFrame and ChainstayBibleBike. Name this new assembly **FrameAssembly.SLDASM**.
2. Reorder the new FrameAssembly to the top of the design tree.

3. Reorder the other parts and assemblies so the bigger assemblies appear higher on the list and the parts appear at the bottom. (Remember that Alt+dragging a component prevents it from being placed into a subassembly.)

4. Drag the part called BB (for Bottom Bracket) into the Frame Assembly (drag without using the Alt key). The assembly FeatureManager at this point is shown in Figure 13.28.

FIGURE 13.28
The starting state and the state as of step 4



5. Select both wheels, and then select Add To New Folder from the RMB menu. Name the new folder **Wheels**, and move it to the bottom of the tree.

6. Expand the Mates folder, select the first four mates, and put them in a new folder (select Add To New Folder from the RMB menu). Name the new folder **Centering Mates**.

   Remember that if you make a mistake, you can go to File ➢ Reload to reload the assembly and parts in the last saved state.

The Bottom Line

Assemblies are more than simply parts and subassemblies put together with mate relationships; several other types of features and placeholders can also exist in the assembly FeatureManager. Organizing assembly components is fairly straightforward and can offer benefits for finding parts as well as controlling suppression and display states globally.

The assembly FeatureManager contains several options for the data to display for subassemblies, parts, configurations, metadata, and features within. Remember that all the data that you include in your SolidWorks documents can be accessed and reused later, so it's worth the effort to name it properly. Descriptions can be very important, both at the part level and also for features and configs.

Master It Use SolidWorks Treehouse to establish the framework for a simple product with two layers of subassemblies. Add some custom properties to the documents that you might want to use to fill in a BOM. Use appropriate templates and save the documents to a special folder.

Master It Speed is a huge part of building an efficient assembly. You measure speed with the Tools ➢ Evaluate ➢ Performance Evaluation tool. Open the Performance Evaluation tool while looking at an assembly (for example, the `Bike Finished.sldasm`) and learn what you can about what types of features and techniques cost you the most rebuild or display regeneration time.

Master It Use the Assembly Visualization tool to sort the parts according to various criteria, and become more familiar with the information presented by Performance Evaluation.