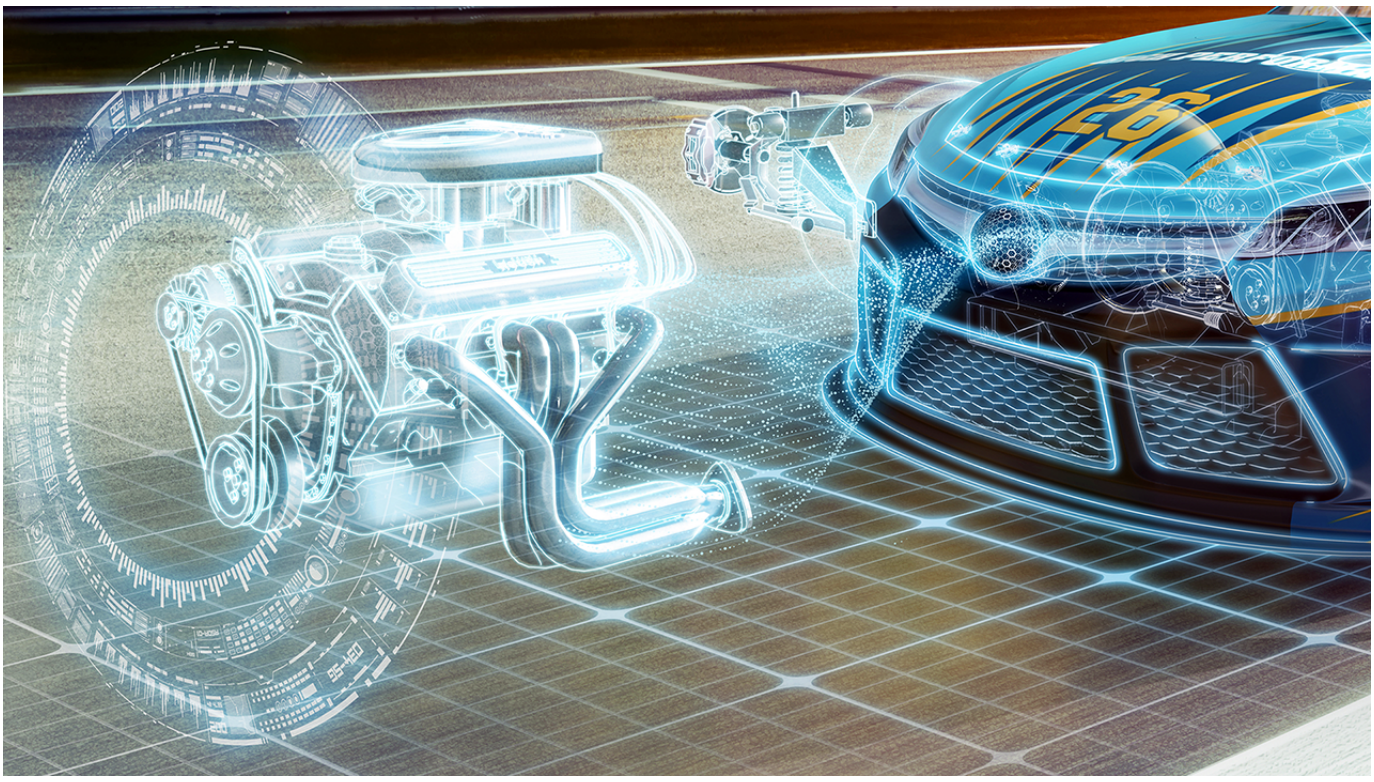
New Math

The Hidden Cost of Swapping CAD Kernels



Schnitger Corporation

When we first wrote about the costs of switching CAD kernels a decade ago, we profiled a company that had twenty years' worth of legacy designs to refresh. They could either find copies of the old software (and the hardware to run it on) or convert the parts to a new format and use a modern CAD system to move the designs forward. Old CAD on old hardware was a non-starter, leaving migrating everything to a new CAD system. But what to convert to? They already used SolidWorks in part of their business and considered moving the legacy parts to that platform. One big problem: Many of SolidWorks' newest features rely on Dassault Systèmes' 3DEXPERIENCE platform. The traditional desktop SolidWorks is built on the Parasolid kernel, while the 3DEXPERIENCE platform uses the CGM kernel. This reliance on two kernels leads many users to worry that building parts in SolidWorks will eventually mean a wholesale conversion from Parasolid to CGM. If you migrate everything today, will you have to do it again in a few years? As you'll see later, converting from one kernel to another can be tricky so, if there is an opportunity to avoid a kernel change, you should investigate this possibility.

The company we wrote about decided that it couldn't afford the risk, disruption, and uncertainty an unclear future might cause. They chose Siemens Solid Edge, which also uses the Parasolid kernel. Sticking with the same kernel simplified moving their Parasolid-based models from one CAD tool to another. At the same time, Solid Edge with Synchronous Technology enabled the team to create usable models from their old parts quickly. How they came to this decision offers valuable insights as you consider your CAD options.

But before we dive into that, what is a modeling kernel? And how could a change from one to another be good or bad for a user?

CAD's Critical Kernel

The objective: model a new component to replace an obsolete element of an assembly. The designer may start with a similar part, modifying only those areas that are unsuited to the new component. He searches his existing part library, finally settling on a specific part because it requires so few changes. He identifies each change, one at a time in the part's history tree, clicks on the dimension to change, inserts the updated dimension, and accepts the change. The component immediately regenerates to show the new size and shape. Changes are displayed as the history tree regenerates with each change. He makes a few other edits and saves the part under a new name.

Each of these actions that display or change the model was executed through the CAD system's geometric kernel -- the CAD application's heart, brains, and engine. Kernels act as the bridge between keyboard, mouse and display, and the computer's processor. The kernel turns complex commands such as "Change the dimension of the flange" into machine-intelligible instructions and collects the result for display, through the application, back to the user.

How does this work? Imagine a car. The driver presses on the gas pedal, feeding gas into the engine. The engine turns this fuel into power; this power drives the gears, pulleys, and electrical system that cause the car to move in response to the driver's commands. If this is a sports car, the driver will get instant acceleration; if not, the acceleration may be more gradual, and the top-end speed is likely to be lower. Drivers have specific expectations for each type of engine and see the engine as a fundamental differentiator between cars.

Figure 1 The engine is the heart of a car, as the kernel is the engine of a CAD system.

The same is true of the geometry kernel in a CAD system. The user creates the instruction to "Extend this flange" (pushes on the gas pedal) and expects a specific result. Using one kernel, "Extend this flange" may be executed through a specific set of geometry and display instructions (gasoline). In contrast, in another, it might use a completely different method for accomplishing the same thing (racing fuel). Both get the job done but in different ways. Just as you cannot quickly put a sports car engine into a minivan, it is challenging to change the geometry kernel that drives a CAD tool. Switching geometry kernels has a significant impact on the CAD software vendor and on customers whose designs are based on the legacy kernel.

Each geometry kernel is designed to satisfy a specific CAD system's requirements and then expanded as those needs evolved. Early kernels were simple geometry tools. Today's kernels are much more sophisticated and use unique algorithms to determine a model's quality and the continuity of surfaces, which will affect how a specific geometric element is processed. While all kernels will enable a CAD designer to change a fillet, each kernel does it differently.

The selection of a geometry kernel is one of the first decisions CAD developers must make. They must decide whether to develop their own or use one of the commercially available alternatives. Many factors are involved in this decision, including cost and functionality, but it is most often based on the CAD product's anticipated use. For example, in what market will this product be used? What must this kernel include to support this market's unique performance characteristics? What interfaces to other products should this CAD tool have?

Commercially available kernels offer the advantage of having a rich feature set and a high degree of reliability; the downside is that the CAD vendor must pay a royalty to the kernel developer for each seat sold and is not in control of the kernel's development. Creating an in-house kernel can take hundreds of working years of implementation and debugging but offers the advantage of total control.

The choice of kernel will directly influence the CAD users' perceptions of their CAD tool's stability, reliability, and performance. Users of CAD products based on the same kernel can more easily share geometry, so Solid Edge and SolidWorks users today can more readily collaborate than, say, users of Dassault Systèmes' CATIA and PTC's Creo. For some CAD customers, this is a critical consideration when planning their IT investments.

A Brief History of CAD Kernels

Kernels started as relatively simple subroutines, written in FORTRAN, to model the faces, edges, and vertices that make up a solid's boundary called a B-rep. B-reps enable the CAD package to calculate attributes like weight, center of gravity, moment of inertia, and other mass properties to accurately and

completely describe the solid. Modelers can combine solids with intersections and unions to represent complex objects. Figure 1 shows how solids are combined to form a CAD model.

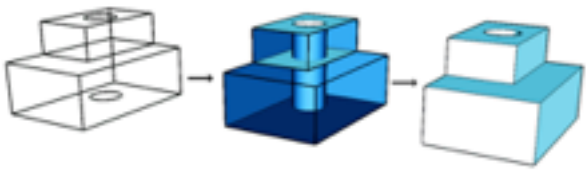


Figure 2 Combining solids to create a CAD model.

The first commercial kernel was Romulus, a package released by Shape Data in 1975. Romulus was a distant ancestor to today's Parasolid kernel, owned and supported by Siemens Digital Industries Software. At about the same time that Romulus became Parasolid in 1985, a team at what is now Dassault Systèmes Spatial began to develop the ACIS kernel that was first released in 1989. Over the years, other kernels have come onto the market, including PTC's commercial offering of the Creo GRANITE kernel and Dassault Systèmes' CGM kernel.

Longevity plays a role in assessing the viability of each commercial kernel. More than 130 software vendors have adopted Siemens' Parasolid. Over 4 million designers and engineers use Parasolid-enabled products and millions more benefit from Parasolid in downstream applications[1]. Spatial says that ACIS is in use in over 350 applications with more than 3 million seats worldwide[2]. PTC's GRANITE, Autodesk's ShapeManager, and Dassault Systèmes' CGM are all newer to the market and have more modest installed base numbers. Dassault Systèmes developed CGM for CATIA V5 and made it commercially available to other developers in 2011; the company estimates that there are several hundred thousand users of CGM-based 3DEXPERIENCE applications[3]. PTC's GRANITE is not a traditional kernel used solely for geometry creation; its interoperability features allow multiple software applications to exchange information without losing data about a model's assembly, associativity, or feature history. PTC created GRANITE to enhance Creo's interoperability in multi-CAD supply chains. ShapeManager was derived from the commercial ACIS kernel and is now proprietary to Autodesk. It is the primary modeling kernel for products such as AutoCAD, Inventor, and Fusion 360, and for interoperability across much of the Autodesk product suite.

There are also open-source (such as Open CASCADE) and academic products on the market, although none has achieved ACIS and Parasolid's widespread adoption. Table 1 shows the most common CAD products and their kernels, as of the date of publication of this paper.

Changing Kernels: Why would a vendor do this?

Over the 40-year history of commercial CAD, quite a few solution providers have swapped out the kernels driving their products. Some did this because they saw limitations in the kernel they were using and wanted to offer capabilities that the kernel would not support, others for financial or reasons of control. But each transition has come at a cost.

Changing a kernel is a complicated endeavor for the software supplier, a lot like swapping out a car engine. It's relatively easy to take out a V6 and replace it with another, but challenging if going from a

[1] https://blogs.sw.siemens.com/plm-components/parasolid-data-exchange/
[2] https://www.spatial.com/products/3d-acis-modeling
[3] https://www.spatial.com/products/cgm

four-cylinder diesel to a V8 gasoline engine or from gasoline to a hybrid gas/electric. In each case, either engine will move the car, but they do it differently.

	Kernel		
CAD Product (Supplier)	***ACIS***	***Parasolid***	***Other***
CATIA V4 and earlier (DS)			Proprietary
CATIA V5/V6 (DS)			CGM
CATIA 3DEXPERIENCE			CGM
Creo (PTC)			GRANITE
Creo Elements/Direct (PTC)			GRANITE
Fusion 360 (Autodesk)			ShapeManager
Inventor (Autodesk)			ShapeManager
NX (Siemens)		✓	
Onshape (PTC)		✓	
Solid Edge (Siemens)		✓	
SolidWorks 2020 and earlier (DS)		✓	
SolidWorks 3DEXPERIENCE (DS)			CGM
Spaceclaim (ANSYS)	✓		

Table 1: Kernels in use in major commercial CAD products

In a software environment, the connections between the engine or kernel and the other systems are called the Application Programming Interface, or API. Changing the kernel requires programmers to change possibly millions of lines of code in the API to ensure that the CAD package makes the correct subroutine calls. For example, one kernel's APIs look at an arc in a clockwise manner, while another uses counterclockwise. Each of these differences must be identified and resolved before a CAD package is functional with a new kernel.

If the CAD package performs as expected after the kernel switch, the development team must create a conversion tool that helps users migrate their part libraries from the old kernel to the new one. This tool is usually a batch process that opens each part in an assembly, recreates it from the history tree, validates that the "after" part is the same as the "before" part, and highlights any areas that need human intervention. Writing this conversion tool is very difficult by all accounts because of differences in how the kernels do rounding and process edges, surfaces, and other basic CAD building blocks, and because of the infinite number of design possibilities.

History shows that even the best converters are likely to succeed with only 90% to 95% of parts, which means that the other 5% to 10% of parts must be manually reconstructed. If 100,000 parts need to be converted, 5,000 to 10,000 would need to be rebuilt, assuming that the new kernel supports this geometry.

From the CAD vendor's perspective, it is simplest to change kernels before commercial customers start using the CAD product. For example, Solid Works was initially developed on the ACIS kernel but switched to Parasolid well before the first customer shipment to resolve performance and functionality issues.

Once a product is in commercial use, the impact on the customer can be significant. In 1998, Solid Edge replaced ACIS with Parasolid in version 5 to improve performance and increase the capabilities the company could offer its customers. It took a year to make the switch even with a large team – and was only possible because the developers had, from the outset, architected Solid Edge not to presume that it would use any particular kernel.

From the customer perspective, the switch also wasn't easy. Solid Edge's developers built a conversion tool but knew it wouldn't translate every part. For that to be possible, every step of the history tree, the recipe for the part, needed to give the same answer in the new kernel as the old. If just one of a thousand steps gives a different answer, the entire history tree must be scrapped, and the part edited or rebuilt manually.

And it's not always obvious what needs to be done: a converted part may look right but not have regenerated correctly, from a mathematical perspective. Or a part may have regenerated differently than the user expected: holes could move, blends might change, and constraints produce another solution. It is difficult to predict which parts will be correct and which will not. This uncertainty leads to a lack of confidence in the overall translation process.

Solid Edge's developers believed then (and now) that the move to Parasolid was the right thing to do, but in the short term, Solid Edge and its customers paid the price for the change.

When Solid Edge changed kernels, relatively few customers were affected because the product was so new. More mature products face problems of an entirely different magnitude. When Dassault Systèmes introduced the CGM kernel with CATIA V5 in 1999, many more customers were affected. Dassault Systèmes had many reasons for changing kernels, including moving from UNIX operating systems with V4 to UNIX and Windows in V5. The V5 kernel was different from V4 (just as Parasolid is different from ACIS). Many users said the conversion was as tricky as moving parts between software packages from two different vendors. Customers were slow to come aboard, and even today. CATIA V4 is still in use in a few long-lived aerospace projects that involve hundreds of thousands of parts, often because of the difficulty of translating between kernels.

Over time, Dassault Systèmes improved the conversion utilities, and service providers stepped in to help with remodeling when needed, but the process was still complicated. In some cases, problems were caused by differences in the way the kernels processed information. Others were created by modeling methods that may have relied on tolerances or other laxities in one kernel that were tightened up in another. Users reported significant data loss in some types of parts and a conversion success rate that ultimately seemed to average about 95%. The remaining 5% of parts required manual rebuilding.

We have one more example to reinforce the 90%-95% success rate of automatic converters. In 2009, Nemetschek Vectorworks changed from its prior kernel to Parasolid. To improve stability and provide greater functionality. The conversion tool that team created was able to convert about 90% of the parts because the old and new geometric kernels made different assumptions as they (re)build features that can affect the way the part regenerates. There's no way to automate a fix for the 5% to 10% of failures; the user must rebuild these parts by hand.

As we've seen, changing a kernel is challenging for both the software developer and the user community and is not something that should be undertaken without careful consideration. Luckily, the lessons from past kernel changes, summarized in the sections below, can help users make decisions and plan transitions.

Changing Kernels: Impact on CAD users

A vendor's decision to change the kernel in a CAD system will affect all users who decide to stay with that CAD tool. Each user team must decide whether staying with a vendor justifies the disruption caused by the change in the "engine" of their CAD tool and look at the relative costs of staying versus switching. The impact on enterprises with many users, high numbers of legacy parts, and more complex processes around their CAD installations will be more significant than those with more modest set-ups. All users, however, will face similar issues:

- Potentially substantial data loss in existing parts, which could number in the thousands – remember that even the best translation tools seem to have a 90% to 95% success rate.
- Possible downtime during parts conversion and lost productivity during retraining on a new CAD tool.
- The expense and time delay caused by hiring an external data cleanup service could be significant if internal resources are insufficient to simultaneously carry out ongoing work and conversion-related tasks.
- Unknown performance and reliability. Arguably the most significant risk in a kernel switch is the transition from a known, familiar tool to an unproven one.
- The timetable for the change and any future development plans to expand legacy and new products.

In the end, a CAD user can only react to the vendor's decision to switch kernels. They have just four alternatives:

1. Stay with the CAD product and plan a transition from the current kernel to the new one.
2. Change to another CAD product that uses the same kernel.
3. Look for the most compelling alternative CAD solution, regardless of kernel and other switching costs, or
4. Do nothing now and wait to see what happens.

It is undeniable that changing kernels creates risk. The question for each user team is how well the benefits of that new kernel balance that risk. Each alternative has pluses and minuses, and their relative weight will be different for each group making the decision -- there is no one right answer for all. Some questions to consider:

- **Does the new version, with the new kernel, offer capabilities that make the switch worthwhile?** In many of the cases discussed above, the kernel change led to core usability enhancements.
- **Does using the new kernel fix problems that are relevant to your work process?** If the enhancements do not apply to your operations, then it may not be beneficial to change.

- **How completely will your specific parts convert to the new kernel?** If your modeling process or most-used features cannot be easily regenerated using the conversion tools, consider the options of rebuilding these parts yourself or hiring an outside firm to do so.
- **How much will it cost to rebuild or remodel all parts that do not convert in a batch process?** If the cost is prohibitive, consider looking at another CAD product that uses the same kernel.
- **How much retraining is required to use the new version of your current product?** How does that compare to the training needed to make your team competent on another CAD tool?
- **What other processes are affected by changes in the CAD kernel?** For example, what third party tools may no longer work because of a kernel incompatibility? What interfaces will not work? How long will it take internal or partner resources to create compatible solutions? How much will this cost? Many third-party products are connected directly through the kernel, but others use their own or neutral file formats.

Deciding to do nothing, the last of the four choices listed above is merely delaying the point at which a crisis will occur. Eventually, the CAD vendor will stop supporting its legacy product, and a process that today can be planned and managed can turn into a fire drill. Delaying the transition to another CAD product or kernel is perfectly reasonable until the CAD team is ready and able to switch -- but all the experts advise: don't put it off too long.

One approach that has seen much traction is the second option, sticking with the same kernel but moving to another CAD product. The company profiled in the first version of this white paper stayed with Parasolid but moved from SolidWorks to Solid Edge. They believed that, while Dassault Systèmes continues to add capabilities that rely on the CGM kernel, Solid Edge developers are pushing the envelope to create new features within Solid Edge on the Parasolid kernel. They benefit from new features without risk. But perhaps the most significant benefit that resulted from their taking action was getting the transition over with and eliminating the uncertainty caused by Solid Works' kernel duality.

Table 2 lays out a conceptual framework for evaluating the costs of a kernel change. Your costs will be different based on your parts' complexity, the automatic conversion tool's capability, how much training your users need, and how many systems are connected to your CAD tool. But it's a way to start putting costs against change alternatives.

You've Decided to Switch Kernels: Now What?

If you opt to stay with your existing CAD vendor through a kernel change, the first step after the decision is planning. Convert all parts at once? When? Or do it in stages? Train users now? Later? Hope they pick it up informally?

Planning a kernel switch is a bit like staging any other major product update's rollout, with the added complexity of ensuring that parts are converted before designers need them.

Many companies have found that it is easier to do this migration all at once, rather than piecemeal. Batch converting the parts ensures that everything is on the same kernel and has been translated using precisely the same settings and methodology for consistency. If all goes well, nothing will need to be converted "on-the-fly."

	Option 1: **Same CAD tool, different kernel**	***Option 2:*** **Same kernel, different CAD**	***Option 3:*** **Different CAD, different kernel**
Tasks	Convert all parts to the new kernel, retrain	No parts conversion, learn new CAD tool	Convert all parts, learn a new CAD
# Parts to convert	100,000	0	100,000
Time to convert using an automated tool	2 minutes/part	0	2 minutes/part
Total tool time	3,000 hrs	3,000 hrs	3,000 hrs
Conversion success rate	90%	100%	90%
Parts to manually edit	10,000	0	10,000
Manual time to rebuild	20 minutes/part	0	20 minutes/part
Total rebuild time	3,000 hrs	0	3,000 hrs
Retraining on new product	1 week/user	2 weeks/user	2 weeks/user
Fixing connections to third party products	TBD	0	TBD
Total time to productivity	3,000 hrs auto-convert + 30,000 hrs manual edit + 40 hrs retraining/user + cost of connections to other products	80 hrs retraining/user	3,000 hrs auto-convert + 30,000 hrs manual edit + 80 hrs retraining/user + cost of connections to other products
The total cost of conversion (depending on labor cost)	**Cost of training** **plus** **Part conversion: $750,000 - $2,000,000**	**Cost of training**	**Cost of training** **plus** **Part conversion: $750,000 - $2,000,000**

Table 2: A framework for considering the cost of migrating CAD kernels

History shows that a conversion tool might successfully migrate only 90% to 95% of parts. The remaining 5% to 10% of parts will require anything from minor hand editing to a significant redesign. The odds of success can be increased by doing a pre-conversion cleanup, choosing the appropriate options (if offered) in the conversion utility, and checking parts at critical points during the conversion itself.

Experts we've interviewed about kernel changes suggest the following for coming through the transition with the least disruption:

- Get as much information as you can, as early as you can, and verify the vendor's claims by speaking with early adopters and reference accounts. Consider, too, what your current supplier and all others you're looking at offer, in addition to basic CAD: is there some feature offered by a CAD supplier that is so attractive that it trumps all other concerns?

- Be sure that you understand the benefits of the new kernel. What processes does your supplier say will be easier? Have reference customers who have made the switch seen these benefits? Test the latest version of the software using your processes and practices. Set up a test environment for a few weeks to verify whether everything works as the vendor promises.
- If your testing proves out the vendor's claims, consider how to upgrade to the new version. Will these benefits make it worthwhile to consider converting your parts? If so,
- Try to understand the differences between the kernels. How do they process self-intersecting surfaces, calculate tolerances, and handle other geometric and topological issues? What parts of your models and your modeling process are pushing the envelope of the new kernel? Will you be able to model these features, for example, in a different way, or will the kernel fail? If the kernel is up to the task,
- Understand how your vendor's part conversion tool works. What settings can you use to maximize success, given your modeling practices? How do they (and you) determine if a part correctly regenerated?
- Is your supplier's support organization geared up to assist? Does the documentation for the kernel and conversion tool help you work through problems? Does the vendor have resources allocated to fix files when things don't go right?
- Recompute your most critical 10, 20, or 30 parts on a test basis and thoroughly check the results. If they don't convert quickly and cleanly, reconsider.
- Proceed with caution since it's challenging to predict conversion failures. Recompute at the lowest possible feature to make sure parts are regenerating correctly.
- Plan the transition very carefully and allocate enough time to rebuild essential parts before production work resumes.
- Check the results of the conversions immediately afterward. Do not file converted parts away for long-term storage without first checking that they converted correctly at all levels. You don't want to figure out why a part doesn't regenerate two years after the changeover with no idea that it is because it failed to convert.
- Convert all current parts at once. You need to minimize confusion between converted and original versions. Keeping on hand only those in the new, correct format will significantly cut down on mistakes.
- Understand how your data management system will interpret the conversion. Will you be assigning new version numbers to converted parts? What does that do to your control scheme? Develop a single strategy and stick to it. Again, errors can creep in if there's uncertainty.

The Bottom Line

A CAD system's geometry kernel is the heart, brains, and engine of the application. Most CAD systems were designed with a specific kernel in mind but can, with time and care, be modified to rely on another kernel for application-to-machine instructions. Each kernel has its strengths, weaknesses, and peculiarities, making a change from one kernel to another difficult on users. In a kernel swap, users must convert part libraries and test connections to third-party programs and other interfaces. Part version control is crucial to highlight that a part is converted but otherwise the same as the legacy part. In the end, each user team must decide which is less disruptive: Converting parts to take advantage of the new kernel's benefits or switching to a different CAD package and keeping the same kernel. Each alternative involves risk but also the potential for great benefit. Moving to a new tool may change your CAD department's trajectory, offering new capabilities and opportunities for growth.

Schnitger Corporation created this white paper at the request of Siemens Digital Industries Software. The analysis and opinions presented here are those of Schnitger Corporation. For more information or to comment, please visit www.schnitgercorp.com. Cover image courtesy of Siemens Digital Industries Software,