Gear Design Software: Engineer Beware?

Gear design software is a wondrous thing. *But watch your back.*

By Jack McGuinn, Senior Editor

When software goes bad, what do we call it? System failure? Human failure? A virus? A number of words will work. How about this? *Glitch.* It has that onomatopoeic quality that fairly screams, "Downtime!" And with good reason — software-generated miscalculations can have very expensive — if not perilous — repercussions. What makes software mistakes particularly sinister is that they may be hard to detect by the QA/Inspection process. You can't put a gage to an algorithm, for example — it's *software*.

Like gears, software is *everywhere*. And like gears, most people never think about the wondrous things that software makes happen around this world on a 24-7 basis. Working seamlessly with gear systems of all types — from the simplest to the most complex — gear software is the friendly "ghost" in the machine that enables robotic surgery, keeps rotocraft rotating, streamlines today's automatic transmissions, keeps gear motor systems running smoothly (and profitably), and ever so much more.

But gears sometimes fail — seemingly for no reason — but we know that's not the case. Software? Same thing. And a design engineer doesn't have to be a closeted Luddite to be a software scoffer. They exist in plain sight. That's one, typically older extreme of the spectrum. And then we have those — typically of a younger vintage — for whom software is the Mothers' Milk of gear design. And finally there is the much larger group — the in-betweeners, we'll call them — who strike that delicate balance of cautious reliance upon software and irreplaceable life/field experience. Is there a right or wrong group? Let's try and find out. To make that happen we talked to a number of people with credentials that qualify them to speak with unquestioned authority to the issues broached in this article. All six of them have authored, co-authored, or overseen the development of their company's proprietary and successful design software programs. As you'll see, they and the companies or consultancies they lead are all *gear knowledge*-intensive. The software expertise comes almost as a value-added premium.

We wondered what, for those designers who cling to software like a favorite stuffed animal, are the most common mistakes that can — and do — occur.

"A permanent source for mistakes is the user himself," said KISSsoft AG CEO Stefan Beermann and president Ulrich Kissling in a joint statement. "He has to enter a large number of parameters; each input can be wrong. Sometimes the meaning of an input is misunderstood. If this happens the program will give correct results, but for another case. So it is essential to make plausibility checks, starting with looking at input and output speed and torque. Checking sense of rotation (important, for instance, if you have a helical gear). And if your final solution is 10 times better than any other in the world, you are a genius, or something is wrong. Usually the latter, no insult meant."

There is no doubt that modern gear software is widely used and very helpful to gear engineers for rating load capacity of existing gear designs or designing new gears. However, calculated results from gear software are accurate only if the input data are correct.

Bob Errichello, president of Geartech, a gear industry consultant, noted AGMA gear failure analysis guru, software developer and *Gear Technology* technical editor, readily concurs with the correct input data concept, adding, "Consequently, it requires an experienced gear engineer to properly formulate the input data. The necessary knowledge and judgment is only gained from years of accumulated experience in designing, manufacturing, and testing gearboxes. The required knowledge includes training in gear materials, heat treatment, gear metrology, gear tribology, gear failure analysis, and bearing technology. Therefore, gear software is a useful tool, but it needs to be used by an experienced gear engineer to obtain meaningful gear designs."

And Mike Fish of Dontyne Systems has perhaps the best answer in terms of why frivolous use of design software can be downright irresponsible. "The most common problem is in believing that the software is a quick-fix or even a replacement for experience

> "(There is) no magic gear design software that is capable (of) delivering a perfectly optimized set of gear parameters by itself, (which) is why gear engineer expertise is essential. (In fact) some gear design programs have a warning 'for gear experts only.' Ignoring this warning is a common mistake."
>
> **Alex Kapelevich, AKGears**

and well-trained personnel within a company."

Indeed, as Alex Kapelevich, president of AKGears and the author of the 2013 release, *Direct Gear Design* (CRC Press), points out, "Software is not magic; it is merely an extension of gear knowledge, fundamentals and experience. "(There is) no magic gear design software that is capable (of) delivering a perfectly optimized set of gear parameters by itself, (which) is why gear engineer expertise is essential. Gear design software is just a tool; it works well in good hands, and vice versa. An engineer that uses gear software should have solid knowledge of gear design fundamentals. (In fact) some gear design programs have a warning "'for gear experts only;'" ignoring this warning is a common mistake."

One more thing on this point: knowing where to start. That may sound obvious. It's not.

Says N.K. "Chinn" Chinnusamy, president of Roscoe, IL-based Excel Gear. "The user should have a clear idea of what he or she is looking for. Incorrect input will result in bad or unreliable output data. Most gear design software available in the market will not tell the user where to start but will simply analyze the data inputted and show results. So the user should know where to start."

And then there's the old truism: software is only as smart as the people that wrote it.

Or is it?

"This is partly true," Fish qualifies, "but it is also only as *beneficial* as the person using it. This includes the psychological barrier of the user departing from a previous procedure for a new one required by the software. The intention is that the new method will benefit long-term in reduced time and more certainty in manufacture (otherwise, why change?), but it is sometimes difficult to demonstrate this to the user if they can't follow the new approach or have a mental block on the principle, perhaps seeing it as a threat. This situation has improved dramatically in the last 20 years with the emergence of much higher computer literacy across the workforce and improved graphics which allow visualization of the gears they have designed."

"Who says that software is only as smart as the person who wrote it?" Beermann and Kissling rejoin. "If the basic concept of a solution is a perfect fit to the problem, often a computer program gets more capabilities than whoever wrote it expects in the beginning. Of course it is also easy and not so rare that a not-so-smart developer spoils even a smart concept."

"Yes, to some extent it is," says Chinn, agreeing with the premise. "But any gear software should be written per standard — either AGMA or DIN or ISO. In this respect it should be the same no matter who writes the software. User-friendly input, clear, concise and easy to interpret output data will depend on the person who wrote it. In some software the user should have knowledge of gear design and metallur-

gy even to attempt to use it, and it will print out several pages of output data which may be confusing to the user if the user is not a gear engineer."

For Kapelevich, it all comes down to ignoring another old truism at one's own risk. "Insufficient knowledge of gear drive application details that result in improperly prepared input data is another common mistake. Every software user should know the rule: 'garbage in, garbage out.' Disregard of this rule results in poor design that may lead to assembly problems, premature gear drive failure or to over-designed product — i.e., greater in size, weight, and more expensive than necessary."

We mentioned at the top that the extent of reliance on the use of gear design software was probably commensurate with the age of the design engineer; and does anyone not know where this is going? With that in mind, to what extent *do* design engineers rely upon gear design software?

"There are basically three groups of engineers," say Beermann-Kissling. "The first group consists of (typically older) experienced engineers that are, or think they are, sophisticated experts in their field. In this group many don't trust any design software at all. Some of them have problems using it, which is related to growing up without any computers at work. The second group uses the design software as a tool wherever appropriate, understands the theory behind it, but doesn't believe it blindly. The third group wouldn't do anything without software. This group mainly consists of young engineers that grew up with computers and, nowadays, smartphones and tablets. In this group there is a certain danger of relying too much on the computer.

"This becomes a problem when the software is used without understanding the theory. In these cases the software has to replace missing knowledge, and that won't work. Referring to the above, you can say that even smart software is lost if it is not used in a smart way."

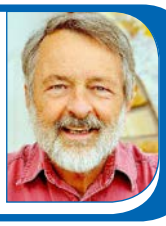
For Dontyne's Fish, it, too, is not always a black-or-white answer; on some occasions an engineer's life/field experience also comes into play, among other factors. And it makes sense — even if it reads a bit like a le Carre *Smiley* novel.

"In our experience we see our customers temper the use of the software with a healthy amount of skepticism and scrutinize the results. Software calculates exactly (or at least should!) to ISO, AGMA, or other standards which may differ from each other or an internal rating system.

"There are known problems with ISO and AGMA that result in differences in interpretation, and it is not unusual for different packages to return different answers for a given gear specification. The standards provide standard methods and not recommended solutions.

"If the software calculates better result than experience, then it is best to go with the worst case scenario until proven by testing. If the software indicates failure but the engineer has experience of similar working gears with acceptable performance, then the application knowledge should be used and relied on. This methodology is consistent with the intent of gear rating standards. This decision process requires an application knowledge base to draw on from the engineer or at least within the company."

> "The gear engineer must be intimately familiar with each of the industry standards to be able to understand why the (gear) ratings differ, and to properly interpret and apply the ratings. It is imperative that gear engineers test software-designed gears to prove that the software is reliable."
>
> **Robert Errichello, Geartech**



Despite all the *Sturm und Drang* about catastrophes-in-waiting, gear-making is, in its way, a forgiving manufacturing process. It's not corneal transplantation. There is no *best* way to make a gear.

But, say Beermann and Kissling, "There are some ways that are not very good. For sure, there are multiple ways to design a gear that are leading to good results in an efficient way. The best basis for a new design is a given design plus some information about the performance of the old gear set. Based on this, it is usually not so hard to find a new variant that fulfills the current requirements. Good design software helps a lot on the way, making proposals and automatically analyzing hundreds or thousands of possible solutions.

"In addition, it is a good idea to go from rough to fine, so start with the macro-parameters like number of teeth, module (or diametral pitch), helix angle, profile shift. Then go down to micro-geometry and further optimize the gear set."

Let's assume Fish agrees: "There is no best way! Have a sure reference point for the application you are making. If this is not available, then there will have to be considerable testing."

And Chinnusamy reminds that "best" should apply to making a gear to its *designed* "size, weight, speed, quality, life, and cost." If that happens, who needs "best"?

What about plastic and PM gears — two hot commodities in play these days — but there's certainly nothing "common" about them. Where does design software fit in their world?

"The properties of these (plastic and PM) are fundamentally different," says Fish. "The standards and models cannot be transposed because it is convenient (though many have tried), since the factors determined by testing are specific to the material and forms originally intended."

As a company, KISSsoft looks at the two materials as two distinct opportunities — but not without challenges. And what's this? — a little "fun" to boot.

"The process of molding gears offers new opportunities and new challenges to the engineer," say Beermann and Kissling. "There is much more freedom in the design, but there are also requirements from the molding process that must be taken into account. Like, for instance, a rounding on the tip that can be produced in a wire erosion process. Then the variety of the materials is huge, and each combination can show significantly different behavior.

The failure modes change — instead of pitting we usually talk about wear. Scuffing will not be a problem, but a complete meltdown of the gear can happen. Temperature becomes crucial for the design. All these challenges make designing plastic gears so much more fun than metal gears (if you don't believe it, try it). The main problem for plastic designs is still the lack of data for material properties. So here it is even more important to have older designs to compare to (also referred to as 'experience').

"Powder metal gears are somewhat in between; more freedom in design, but a material which is close to classic steel with its properties.

"In both cases, it is recommended to use a definition of the tooth form which is closest to the generating process. In this way you avoid a lot of possible errors in the design; like, for instance, collision with the root fillet. If you want to define your tooth shape directly, make sure you check the proper meshing of the gear set afterwards."

"Who says that software is only as smart as the person who wrote it? Of course it is also easy — and not so rare — that a not-so-smart developer spoils even a smart concept."
Stefan Beermann and Ulrich Kissling, KISSsoft AG

Aside from fielding the questions put to them for this article, there is no lack of fodder regarding software-induced — or enabled — errors in gear manufacture. So we pressed for a bit more reaction about software performing unpredictably in circumstances beyond its design parameters.

"If you've ever used a pair of pliers for nailing, this is not coming as a surprise," quip Beermann and Kissling. "A good program should inform the user if he uses the software outside of its well-defined limits. Still, if used carefully, this is not really a problem. It is important to assess the result and check it thoroughly.

"And there will be cases when only a prototype test will give the final answer. But we are talking about the design process here, not controlling a nuclear power plant. So, the worst that can happen is that the design doesn't work in the end. The target of design (and even more analysis) software is not to make prototype testing obsolete; the target is to reduce the number of tests needed."

At Dontyne, Fish points out that, among other things, software is incapable of making a decision — remember, garbage in, etc. "The software should only be an aid for the engineer and used within its bounds. If the engineer recognizes a problem in the calculations it should be reported to the publisher and it will be corrected. If errors occur, then engineers may be trying to apply the calculations for something for which it is not intended."

KISSsoft's team is right with Fish on this one. "You mean except for quan-

"In some software the user should have knowledge of gear design and metallurgy even to attempt to use it, and it will print out several pages of output data which may be confusing to the user if the user is not a gear engineer."

N.K. "Chinn" Chinnusamy, Excel Gear

tum flux? Design software is not thinking, it is following algorithms. Those algorithms might run into dead ends, or simply be not appropriate. But this is a question of software design. We tend to believe that the software should do the stupid part of the design work, proposing and evaluating solutions and do the number crunching. The engineer using it should make the decisions.

"Let me ask you back: What do you think makes the engineer think unpredictably?"

Adds Fish, "There may also be fundamental problems with the approach. FE methods are used increasingly, but can generate a different result depending upon the boundary conditions, method used, and element types. This is confusing and looks unpredictable, but is not strictly a problem with the software. The models should have a thorough testing program behind them to show they are appropriate to the situation. It is important that those who write gear standards provide enough numerical examples to allow test programmers to test their coding. Gear trade bodies can also minimize this risk by arranging informal round robin comparison exercises that provide additional examples to verify software."

Errichello points out that, "A factor often overlooked is that gear software must be validated. This is not a trivial issue, and it takes literally years of testing to determine whether particular software is free of bugs and gives valid results. Typically, a gear engineer tests a candidate software against an archive of example gearsets that were calculated from other trusted software or hand-calculated examples.

"Some software has capability to rate gears according to AGMA, ISO, and DIN industry standards. However, the rated capacity for a given gearset will differ for each of these standards. The gear engineer must be intimately familiar with each of the industry standards to be able to understand why the ratings differ and to properly interpret and apply the ratings. Finally, it is imperative that gear engineers test software-designed gears to prove that the software is reliable."

For more information:

AKGears LLC
Phone: (651) 308-8899
ak@akgears.com

Dontyne Systems
Phone: +441912064021
info@dontynesystems.com

Excel Gear
Phone: (815) 623-3414
quotations@excelgear.com

Geartech
Phone: (406) 266-4624
geartech@mt.net

KISSsoft AG
Phone: +41 55 254 20 50
info@KISSsoft.AG