Published in PLAN, April 2011, page 38

By Anne-Marie Gagnon Eng., in collaboration with Jocelyne Hébert

Software, a wonderful tool… of which to be weary!

Analysis and design software are commonly found in today's businesses and are often integrated in complex systems covering their entire operations. Software, without which we can no longer live, remains a tool that, when ill designed or wrongly used, can cause serious errors for which engineers can be held responsible.

Here is an example of engineers asked to examine the hull of a boat using finite element analysis. To be on the safe side, they used two types of software to yield solutions. Results are similar overall; however, some differences appear in an area where there are more constraints. On further analysis, the engineers noticed a weakness in a type of elements in one of the software. If the results hadn't been thoroughly verified, cracks could have appeared quite rapidly at that particular spot in the hull!

NO SOFTWARE BEARS THE TITLE ENG.

When engineers use software, and computers in general, they are held to the highest standard of caution. Software are not perfect simply because they are easy to use, fast and highly performing. A software programme is a tool and should never be substituted for the engineer's thought process, judgment and sense of responsibility.

In other words, as is the case with every tool they use, engineers are responsible for the work they carry out with the help of computer software. Moreover, it is their duty to validate the results they obtain. They bear this responsibility regardless of the complexity of the problems or the tools used to solve them. Even when faced with a challenging deadline or hefty competition, an engineer cannot rely solely on results obtained through software. Should errors arise, the engineer cannot cast blame on the tool.

In fact, engineers should only use computer software when they already have an idea of the results. Before carrying out calculations with a computer programme, the engineer should at least be able to pinpoint the order of magnitude of the expected results.

Engineers should also have extensive knowledge of the software they intend to use. Those who do not have the required expertise in the application domain of a given software are taking serious chances, not to mention the fact that they are violating section 2.04 of the Code of ethics of engineers:

"The engineer shall express his opinion on matters dealing with engineering only if such opinion is based on sufficient knowledge and honest convictions."

The computer and software work as intermediates between the physics model and its underlying assumptions.

If the work includes designing a software, errors could occur at this stage.

Engineers who design software are responsible for the product they develop, much like civil engineers are responsible for designing reliable structures. Their skills and qualifications are subject to the same criteria as those of engineers who practise in other fields.

Finally, engineers must verify the results until such time as they are absolutely sure of their validity, considering all underlying assumptions. In short, they must be weary of this tool's effectiveness, even if they chose this particular tool specifically for its effectiveness …

MAIN CAUSES OF ERROR

In order to obtain good results, we must first choose quality and appropriate software. Such quality should not be established after the fact, but long before and while the project is being carried out, starting with an objective evaluation of the products available on the market. One of the best methods is to test the software using actual problems. These tests need not be complex in order to thoroughly assess the limits of the software and validate typical results. In general, software programmes that are well known and widely used are more reliable, but may nonetheless have certain shortcomings of which we must be aware.

Engineers must determine the limitations with respect to the validity of the software they intend to use. The computer and software work as intermediates between the physics model and its underlying assumptions: if the limits of the simulated model are not well defined, then the engineer is liable to lose track of the model and its hypotheses and might forget information that is essential in validating the results. One must have extensive expertise in the field of application as well as in computer-based modelling so as to avoid this trap. We must always keep the limitations of the software in mind, constantly question the model and make sure the information obtained is correct.

Applying software to cases that arise infrequently may also cause errors. If one of the results is a division by zero, the computer may stop or worse, it may carry on with the calculations and yield wrong results. Such a situation may arise when software is used routinely over a long period of time. If the person who has developed and perfected the programme is no longer available, the original assumptions, the limitations and the methodology may long be forgotten.

HOW TO VALIDATE RESULTS

Even if all engineering fields readily resort to computer science, it is difficult to come up with a general methodology which can be used to validate the results.
Here are a few guidelines:

1. Make sure that the input data is valid; this is all the more important given that most software only perform a summary evaluation of the data entered, thereby detecting only the most obvious formatting errors.
2. Have a good knowledge of the software's scope of application and the assumptions set forth.
3. Verify the results, namely through a visual exam. Curves and graphs will quickly establish trends and validate orders of magnitude. Engineers can also make sure that the overall laws relating to the system's equilibrium and flow are respected. For example, the sum of forces and torque applied to a fixed structure must always amount to zero, regardless of its complexity.

By following these few steps, engineers can avoid many mistakes. They also keep a mindful eye on the tool they use, an attitude we would do well to adopt when using highly performing, yet imperfect, technologies.