



Narrowing in on spreadsheets

Spreadsheets are great, but do you know their limitations?

The pharmaceutical sector is a billion dollar industry with a huge responsibility towards its customers and investors. The main tools for fulfilment of this responsibility are ensurance of compliance and maintenance of control. It is a time-consuming job to uphold these responsibilities, and many important decisions regarding this subject are taken every day. It is important to make carefully considered decisions and follow them up. It is also essential to stop once in a while and reconsider their validity and relevance.

One of the decisions should relate to spreadsheet usage. Spreadsheet programs such as Excel are commonly accepted by the pharmaceutical industry and many spreadsheets are developed and used for a multitude of purposes.¹ This article will focus on the hazards of using spreadsheets in a good manufacturing practice (GMP) environment.^{2,3}

All examples and references used in this article are

based on Excel because it is the most commonly used spreadsheet program.

Anders Keldsen

Spreadsheets are very easy to use and people with basic skills in software development can achieve a lot with little effort. This is the greatest advantage of using spreadsheets, but also the greatest hazard. Because they are easy to use and many people have rudimentary knowledge of them, it is often more cost effective for a small unit to use spreadsheets to create automated calculations instead of developing a software application.

The following scenario is very common. A small unit has a spreadsheet developed and validated. The spreadsheet is satisfactory and soon several other small spreadsheets are created for various purposes. They become increasingly complex and contain more and more logic for the unit's normal operations. It will still be the same employee who developed the original

version that is now updating and maintaining the complex spreadsheets. In most cases, that employee will lack sufficient training in software development to implement, for example, proper error handling.

The real challenge when using spreadsheets is to determine when to use them and when to employ a proper software application. This article will define such occasions — all from a GMP point of view.

When to use spreadsheets

Spreadsheets were meant to be used for calculations — not for handling complex logic. Simply put, if a problem cannot be expressed with mathematical formulae, do not use a spreadsheet.

An example of a problem suitable for spreadsheets can be seen in Equation 1. This is a mathematical formula and is purely based on calculations.

$$p = (x_{top}^2 - x_{bottom}^2) + kx \quad (1)$$

An example of a problem not appropriate for spreadsheets is outlined in Table 1.

Table 1 has conditions that dictate how the calculation should be performed. Conditions such as these can be handled in a spreadsheet, but tend to become quite complicated and error prone. Therefore, some type of error handling must be implemented. What happens if equipment A is used but the user does not specify any equipment in the spreadsheet? What about the date calculation that must

be performed to check the remaining time until maintenance for equipment C — can that calculation be trusted?

In the most widely used spreadsheet applications the date format applied to the spreadsheet is based on that used on the computer running the spreadsheet. Because date formats can vary (dd/mm/yyyy, mm/dd/yy, yyyy-mm-dd etc.) depending on the settings in the operating system, the spreadsheet developer has to take care of error handling regarding different date formats. This can prove to be very challenging, particularly for those with little training.

When using conditions in a spreadsheet, another problem arises, namely language differences. To illustrate this, a simple example is provided in Table 2:

The code written for cell A2 in an English version of Excel would be:

```
IF(A1 1,"One",
IF(A1 2,"Two",""))
```

And in a Danish version of Excel the same formula would look like:

```
HVIS(A1 1,"One";
HVIS(A1 2;"Two"))
```

Notice the two differences:

- in Danish "HVIS" is used, in English "IF" is used
- in the English version commas are used, whereas in the Danish version semicolons are used.

This means that if a Danish company's IT department decides, at a corporate level, to switch from a Danish version of Excel to an English version, the small unit's spreadsheets would no longer work.

The language difference outlines one of the problems with making a program depend on systems controlled by a central instance. That central instance (IT Department) could also make other decisions that would have a significant impact on the spreadsheets. They could choose to change from using Excel to Lotus 123 or OpenOffice Calculator, or just update Excel from Excel97 to Excel 2003. The only way to determine the level of impact is to make a risk assessment, which at worst results in a complete revalidation of the

spreadsheets. Because spreadsheets are dependent on Excel, and because Excel is normally controlled by the IT department the units using spreadsheets do not have full control of their applications — they are at the mercy of the IT department. This dependency can be avoided if a proper software application is developed.

When a unit has developed several spreadsheets for various purposes, they might want to combine these, or even better, make them communicate. Spreadsheet 1 calculates a value that is used in spreadsheet 2, so why not transfer that value automatically? (Figure 1). This can be done using Excel, but the following scenarios must be considered and taken care of with proper error handling:

- Spreadsheet 2 does not exist. It could have been renamed or moved.
- The user does not have read access to spreadsheet 2.
- Spreadsheet 2 was updated so the value is not located in the same position. This means that spreadsheet 1 will retrieve the wrong value and base its calculations on incorrect data.

The first two scenarios can be handled properly with error handling, but the third cannot necessarily be detected by spreadsheet 1. Therefore, when updating spreadsheet 2, greater effort is required from the developers to either maintain the same layout in the spreadsheet or remember to update all spreadsheets using spreadsheet 2 — if not, undetectable calculation errors could result. Because of this it is also undesirable to use Excel for storing data referenced by other programs; a database should be used for this.

Security

When creating spreadsheets some measurements should be taken to maintain control over them. In a regular spreadsheet, every user with sufficient access can change the content. To avoid this, some simple measures to take include:

- Save the spreadsheet as an Excel template. This will prevent users accidentally overwriting it through Excel.
- Use write-protection on all cells the spreadsheet user does not interact with.

Table 1 A problem that is unsuitable for spreadsheets.

Equipment A has a maintenance period of 30 days, equipment B of 10 days and equipment C of 1 year.

When equipment A is used, calculate using formula 1.

When equipment B is used, use formula 2.

When equipment C is used, use formula 2 except if there is less than a month for calibration, then use formula 1.

Table 2 Language in spreadsheets.

If cell A1 is 1, then display "One" in cell A2.

If cell A1 is 2, then display "Two" in cell A2.

- Save the spreadsheet in a central data storage and restrict all users to read-access only.
- Protect the Visual Basic code with a password.
- Update all passwords for the spreadsheet regularly.

Because spreadsheets are vulnerable, perform a regular control of the content to minimize the damage caused if an unintended change was made. Another measure that can be taken is to educate spreadsheet users to know where to expect errors — this

will provide a better chance of discovering the error.

Summary

This article has highlighted how vulnerable spreadsheets are compared with a proper software application. There are three main areas of weakness:

- Because spreadsheets are easy to develop, people with little software development training will often perform this task and, therefore, insufficient error handling may be present.

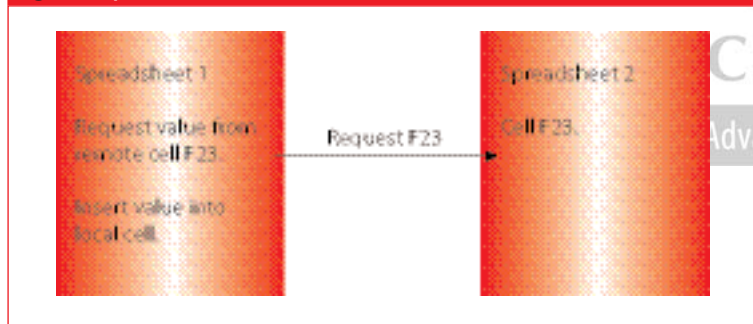
- They depend on Excel (or a similar spreadsheet application) that is most commonly controlled by a central instance.

- They can still be changed (intentionally and unintentionally) after development. Special measures must be taken to prevent this.

Even if spreadsheets are developed carefully and are fully validated, it is important to maintain compliance. This means a proper change control system and maintaining baselines.⁴

For some interesting reading regarding spreadsheets, try the very spreadsheet-sceptical site.³

Figure 1 Spreadsheet communication.



References

1. FDA: Spreadsheet design and validation Part 1 and 2
2. www.21cfrpart11compliance.com/VCS/evcs01.htm
3. European Spreadsheet Risks Interest Group www.eusprig.org
4. C. Stage, *Pharm. Technol. Eur.* **18**(2), 16–18 (2006). [PTE](#)

Anders Keldsen

is a project manager at QAtor, Denmark.