ACOUSTICS 2017 PERTH
SOUND, SCIENCE AND SOCIETY

An Open Source Noise Calculation Toolkit for Acoustics

Philip Setton(1)

(1) Acoustics, WSP, Melbourne, Australia

ABSTRACT

Many Acoustic Engineers today use custom spreadsheets or custom functions to perform their calculations to further enhance the speed and reliability of their calculations. This offers benefits over manually entering tables of data or writing formulas explicitly, both of which can result in human or syntax error.

Proprietary software packages have also been developed to perform acoustics calculations. The ongoing development of these packages occurs at a slow pace. Such packages may also encourage a supplier of products whose specification data is included within the software.

'Open source' software refers to any software with editable source code. Open source software offers benefits over proprietary software as new elements can be appended to an existing platform, documentation can be shared for the tool and each piece of code can be checked to ensure it is functioning as intended and documented.

In this paper, a new open source acoustics calculation platform, *Trace*, is presented. The platform is built as a Microsoft Excel Add-In and contains tools for common acoustics calculations, mainly centred around mechanical noise, simplified environmental noise, and basic room acoustics.

A collaborative approach to software development and quality assurance will provide the Acoustics Engineering community a more transparent and flexible system than those currently available. By 'crowd-sourcing' the problem, development will not rely on a single person or entity. The most successful platforms are those with a strong community whose collective expertise can be utilised – such a model can be adopted for acoustics engineering for the benefit of all users.

1 INTRODUCTION

An anonymous Acoustic Engineer once said "any calculation that can't be done on a napkin probably isn't worth doing". An interesting idea, which led to an interesting counterpoint: Any calculation that doesn't follow this rule should probably be done on a computer.

A number of software tools have been developed to assist in acoustic calculations. These software tools are highly varied in structure and feature, and has been developed on a variety of platforms. They may be developed as separate programs or appended onto existing software (Add-Ins/Plug-ins). Software tools are often developed by individual engineers for use internally within an organisation. They may also be published by product suppliers with a library of product data within the software used to incentivise the users to purchase products from the software developer, based on their products' acoustic performance. The underlying motives of the developer in creating and maintaining the software should therefore be subject to some scrutiny. The longevity of such a project is also at risk, should the product supplier cease to support the software, or go out of business. Major improvements to such software packages are often slow to roll out. Bug fixes may come more regularly, however determining the root cause of issues is very difficult for the user when the source code is hidden.

The following paper presents a new calculation platform, *Trace*, built as an Excel Add-In. *Trace* has been developed to assist in acoustics calculations, centred particularly on mechanical noise and room acoustics. The benefits of open source software are discussed and a case is made for its application to Acoustics.

2 'OPEN SOURCE'

2.1 History

The term 'Open source' has been in use since 1997 when the term was used to describe an alternative development previously referred to as 'free software', partially as was felt the old term discouraged business adoption (Raymond, 2007). Proponents of open source software refer to a 'bazaar model', where the decentralisation of development can be likened to "a great babbling bazaar of differing agendas and approaches" (Raymond, 1999).

The development model for Open Source Software (OSS) generally follows the following principles:

- Users should be treated as co-developers – users are encouraged to submit additions to the software, code fixes, bug reports, documentation etc.
- Early releases – the first versions should be made available as soon as possible to increase the chances of finding co-developers early.
- Frequent integration – some large projects may even build nightly automatically to avoid the overhead of fixing a large number of bugs at the end of the project lifecycle.
- Several versions – by having an older, stable version and a newer, buggier version, users can choose if they want to see new untested features first, even if they may be unstable.
- High modularization – this allows development to occur in parallel.
- Dynamic decision making structure – decisions with respect to implementation of changes are still required, these structures may be formal or informal in nature.

These principles are generally applied to the development of *Trace*, with a long-term goal to aligning with all the principles as the project continues.

It should be noted that the original OSS model as proposed by Raymond is not without its critics - some argue that OSS should instead be considered as a special case of academic research (Bezroukov 1999). These criticisms notwithstanding, the broad principles of OSS as discussed are considered applicable, or at the very least, aspirational.

2.2 Open source software today

There are currently many OSS projects suitable for different applications including: word processing (LibreOffice); engineering calculations (Octave); CAD (FreeCAD); image processing (GIMP); video playback (VLC); audio editing (Audacity); 3D acoustic ray tracing (I-Simpa (Ifsttar 2016)); and many others. OSS projects have also been shown to be successful for niche areas of interest, with collaborators coming together, regardless of distance from each other. The development model of open source projects also means that features may be developed and a different version of the software arise, without any direct control by the original authors.

Many studies have been conducted regarding OSS, particularly with regard to the motivations of OSS developers (Hars 2001), (Oreg and Nov 2008). The number of developers per project has been found to roughly follow a power-law distribution (Madey, Freeh and Tynan 2002), although the contributions may be skewed towards a smaller sub-group of developers. An analysis of open source projects (Ghosh, et al. 2002) showed that 74% of code was written by the most active 10% of authors.

2.3 Reinventing the wheel

Almost every acoustic engineer working today will currently use spreadsheets to perform routine calculations. These are usually centralised and shared within an organisation, although a local copy is often made by individuals who have their own improvements on the centralised version. Some firms may also use custom functions for some computationally difficult but ultimately not advanced tasks. A common example is a function to logarithmically sum input values, the mathematics of which is not very advanced but writing out may take some time.

Given the propensity of each individual firm to develop tools as they see fit, the question then arises as to the efficiency of sequestering these tools within a given organisation. Any time spent developing the tools will quickly pay for itself with repeated uses, however duplication of the development process can only be dealt with by a larger group of developers, whose collective efforts aim to strengthen the platform.

By combining the existing resources of the acoustic engineering community, many features can quickly be added to the tool, and would most likely be added on an as-needs basis. Some minor modifications to existing users local code would be required to conform to the structure of *Trace* but the underlying logic and method of the functions (which is the element that takes the most time to develop) would already be established.

2.4 Quality & Documentation

Ascertaining the method by which the code performs a calculation is a task that requires focus and attention to detail. A Quality Assurance check of all procedures within a software package is an immense task requiring many hours of attention and care. Evaluating the quality of proprietary software is practically impossible as the source code cannot be seen. For example, it is possible a procedure may give correct answers within a typical range of values but give erroneous values outside the normal ranges. This is especially true where computer data variable types must be defined at the outset. Such a problem may not result in an explicit error but may change the final result.

The task of Quality Assurance checking and documentation are ideally suited to be 'crowdsourced'. The user/developers should follow good practice in commenting and annotating the code to explain to others how the procedure is undertaken. If the instructions are unclear or incomplete, other users may seek to clarify or enhance the explanation. Establishing a repository for documentation, as an online help tool, has been successful for many other platforms - such an approach will be undertaken for *Trace*.

2.5 Application for Acoustics

The research available to the acoustics community is quite substantial, given its relatively small size. Conferences and other industry events can assist in the sharing of methods and approaches; the 'cross-pollination' of staff between companies also leads to knowledge sharing. This 'knowledge' is not limited to those resources that can be stored electronically and transported instantaneously – we should consider that "knowledge includes subjective and personal assessments of the value, meaning and use of information" (Druguid 2001). The ability to quickly formulate calculations does not assist in the understanding of the underlying acoustics fundamentals. The development of an open-source platform should therefore be viewed as an opportunity for collaboration between firms without providing a competitive advantage to any one company or group.

Importantly, the formulas, functions and tools used in *Trace* can be found in any reasonable textbook on acoustics. They can be checked against the original and examined by any user. Many proprietary software platforms hide the underlying mathematics within source code, even when the formulas/relationships are well known. This limits the understanding of the user for no real benefit, except the increased reliance on the software platform itself. Should the software cease to be supported in the future, the entire history of calculations undertaken with that software would cease to be accessible in any meaningful way. Static printouts or PDF versions would not constitute a complete understanding of the calculation, even with a reasonable level of documentation. By allowing access to the source code, the underlying functionality cannot be lost, even if ongoing development stalls. Of course, this is not the intention of the project, but is an issue worth consideration from a risk minimisation standpoint.

Additionally, it is the opinion of the author that the training opportunities for young acoustics professionals are easier in an open source environment than proprietary software. Younger engineers who have a skill set that includes coding could then supplement the platform and enhance their learning at the same time. More experienced coders would only be required to oversee the code development process. For example, implementation of a custom function for a new version of the standard could be set as a task, the development of which requires the learner to understand the standard and implement it in a general way. The process would then be open to checking by an engineer with more experience.

3 FEATURES

3.1 Design elements

The underlying design elements are intended to make calculations as simple and flexible as possible. Microsoft Excel has a range of in-built features, which usually allows for many different methods to solve a problem. Establishing the design elements of the platform early in the development should assist in the continued growth of the platform.

The laws for *Trace* are as follows:

- Where possible, the full equation is to be written out, except where:
  - Empirical relationships exist / lookup tables are required; or
  - the full equation is difficult to read as an inline function.
- User input cells is to be 'post-it' yellow RGB (251, 251, 143).
- Final answer is be in 'friendly' blue RGB (146, 205, 220).
- Units (10 m^2,Q=2) are presented as custom formatting with the cell.
- All losses are to be shown as negative numbers.
- Blank Calculation Sheets and Standard Calculation Sheets are to be printable with a layout suitable for fitting to standard A4 sheets (portrait or landscape, as appropriate).
- All sheets are to have a standard header block layout (see Section 3.2).

$$SPL_{Total} = 10\log\left(10^{\frac{SPL_1}{10}} + 10^{\frac{SPL_2}{10}} + 10^{\frac{SPL_3}{10}} + \cdots + 10^{\frac{SPL_9}{10}}\right) \quad (1)$$

The above inline formula (equation 1) would be prohibitively long and difficult to check for Quality Assurance (QA) purposes. For this reason the function *SPLSUM* has been created, meaning the inline formula can be written as:

SPLSUM(E9:M9) (2)

Calculations are generally only auto-filled using the Ribbon Controls for the templates provided but can be called from any sheet by typing '=' and the function name.

3.2 Calculation Sheets

The platform makes a clear distinction between Blank Calculation Sheets and Standard Calculation Sheets. Blank Calculation Sheets and Standard Calculation Sheets have a common title block layout, as shown in Figure 1. The title block contains project information, the date, as well as sheet and revision number. All sheets are titled with their version in the file name. Details about what was updated will be shown in a comment on Cell A1.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1		**Project No.**	12345						**Date**	02 July 2017				**Sheet**	1
2		**Project Title**	123 Fake St						**Engineer**		N. Gineer			**Rev**	1
3		**Description**	Example calculation											**Type**	OCT

Figure 1: Example header block

	Description / Comment	Weighting		Octave Band Centre Frequency, Hz									Parameter
		Lin	A	31.5	63	125	250	500	1000	2000	4000	8000	
12	A Weighting			-39.4	-26.2	-16.1	-8.6	-3.2	0.0	1.2	1.0	-1.1	
13	SWL of source	91.8	**74.9**	60.0	91.0	82.0	77.5	73.0	66.5	61.0	55.0	50.0	
14	Number of sources			3	3	3	3	3	3	3	3	3	N = 2
15	Total SWL from Source	94.8	**77.9**	63	94	85	81	76	70	64	58	53	
16	Direct Path Divergence			-45	-45	-45	-45	-45	-45	-45	-45	-45	
17	Direct SPL	49.8	**32.9**	18	49	40	36	31	25	19	13	8	
18	Barrier Attenuation - Path 1			-6	-7	-8	-10	-13	-16	-19	-22	-25	
19	Path 1 (d1>d2)	42.5	**21.9**	12	42	32	25	18	9	0	-9	-17	
20	Barrier Attenuation - Path 2			-8	-11	-13	-16	-19	-22	-25	-28	-31	
21	Path 2 (d3>d4>d2)	38.8	**17.0**	10	38	27	19	12	2	-6	-15	-23	
22	Barrier Attenuation - Path 3			-19	-22	-25	-28	-31	-34	-37	-40	-43	
23	Path 3 (d1>d5>d6)	27.5	**0.9**	-1	27	15	8	0	-9	-18	-27	-35	
24	Barrier Attenuation - Path 4			-19	-22	-25	-28	-31	-34	-37	-40	-43	
25	Path 4 (d3>d4>d5>d6)	27.3	**0.8**	-1	27	15	7	0	-10	-18	-27	-35	
24	**Total SPL**	44.3	**23.3**	**14**	**44**	**33**	**26**	**19**	**10**	**1**	**-8**	**-16**	
25	Derived Insertion Loss	-5.5	**-9.6**	-3.8	-5.2	-7.0	-9.3	-12.0	-14.9	-17.9	-21.0	-24.0	

Figure 2: Standard calculation sheet cell styles

3.2.1 Blank Calculation Sheets

Blank Calculation Sheets are laid out for the purposes of setting up calculations from scratch. There are currently four such sheets:

- Octave Band (Linear Input) – 31.5Hz to 8kHz bands
- Octave Band (A-weighted input) – 31.5Hz to 8kHz bands
- One-Third Octave Band (Linear Input) – 50Hz to 5kHz bands
- One-Third Octave Band (A-weighted Input) – 50Hz to 5kHz bands

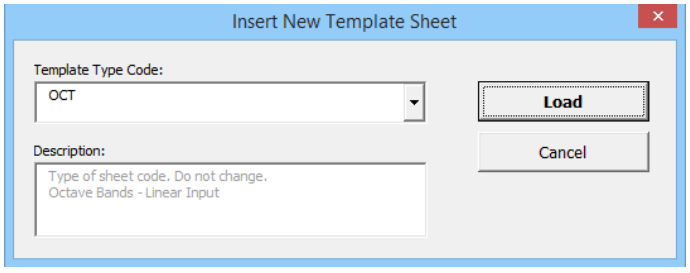

Figure 3: Blank Calculation Sheet selection form

There are no formulas in Blank Calculation Sheets, except for a running linear and A-weighted total in each line. Users can utilise the in-built functions to undertake a calculation – this may be supplemented by any equation in the usual manner. Additional sheets are added to the current workbook. The named range TYPECODE denotes the type of Blank Calculation Sheet – this is fed into other logic structures within the code.

3.3 Standard Calculation Sheets

These sheets are set up to calculate common acoustic problems. The 'Standard Calculation' button on the ribbon scans the directory (*Trace/Standard Calc Sheets*) and populates a list in a form. The user selects from the list and is prompted to save a copy, which is date stamped by default.

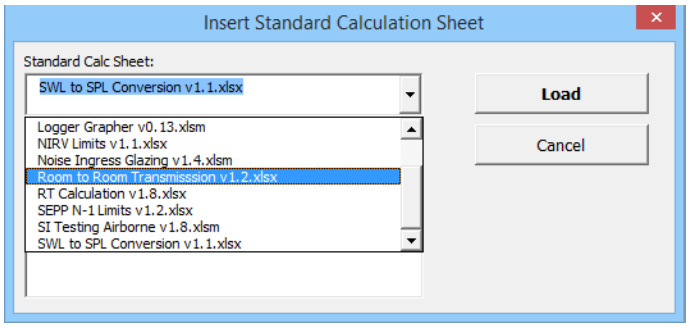

Figure 4: Standard Calculation Sheet selection form

As with Blank Calculation Sheets, the Standard Calculation Sheets are laid out to fit a on a page when printed. Not all lines of the calculation are displayed in the final presented sheet, but should be sufficient for a review. Table 1 shows a list of the Sheets that have been developed to date. Other sheets can be developed and shared within an organisation by simply copying them into the appropriate folder, however sharing calculation sheets to the entire development pool for common acoustic calculations is strongly encouraged.

Table 1: Standard Calculation Sheets (Currently available)

CODE	Name	Description	Latest Version
BA	Barrier Attenuation	Implements Maekawa's formula to predict barrier insertion loss	1.3
GLZ	Noise Ingress Glazing	Predicts internal noise level of external noise source through glazing in one-third octave bands	1.2
LG	Logger Grapher	Presents ARL316 logger data	1.1
N1L	SEPP N-1 Limits	Determines Noise Limits under State Environment Protection Policy No N-1 - Control of Noise from commerce, industry and trade	1.1
NR1L	NIRV Limits	Determines Recommended Maximum Noise Levels (RMNLs) under EPA Publication 1411 - Noise in Regional Victoria	1.0
P2W	SWL to SPL Conversion	Converts SPL to SWL for up to four sources. Converts SWL to SPL based on RT of room	1.0
R2R	Room to Room Transmission	Predicts noise levels in a room, transmitted through a partition from another room	1.0
RT	RT Calculation	Estimates Reverberation Time using 3 methods based on room geometry and finishes.	1.6
SIA	Sound Insulation - Airborne	Calculates Dw and DnTw for measured wall/floor systems	1.4

3.3.1 Functions

The in-built functions are currently centred around mechanical noise calculations and room acoustics. This could easily be extended as the users inform the direction of the development. There are also two columns in Blank Calculation Sheets (which may be merged into a single cell in some cases) to allow for certain variables to appear as user inputs within the spreadsheet, instead of being 'hard coded' into the formula. As an example, the functions GetASHRAEDuct has the following structure:

GetASHRAE(freq As String, L As Long, W As Long, DuctType As String, Distance As Double) (3)

By default, the function would input the variables *freq*, *DuctType* and *Distance* as cell references; the variables *L* and *W* (for length and width) as fixed numbers. All variables are set via a custom form, but can be changed manually by the user at any time. For special codes, such as *DuctType*, data validation is applied to the cell to assist in quickly changing variables with few options.

3.4 Ribbon Controls

Buttons on the Excel Ribbon can be added using Custom UI Editor (dmahugh 2006). The UI editor enables adding buttons, groups, pull out menus and other controls using custom XML (extended markup language). Button icons are added from the library available within Microsoft Office (Microsoft 2006). Icons for the existing tool have been selected to match the physical or conceptual form of the function being inserted (where possible). Figure 5 shows the current layout of the Ribbon.

Figure 5: Ribbon Controls (Version 1.13)

3.5 Program Structure

Creation of a toolkit requires the code to be structured and labelled as clearly as possible. As changes to the code can be made by any user/developer, establishing naming conventions and program structure from the outset should 'lay the groundwork for all future additions. Figure 6 shows the elements of the Add-In and their connections. A dummy function *CustomFunction* is used to demonstrate the flow of each element of the code. In general, the function is located in a module with the same name as the tab group on the Ribbon.

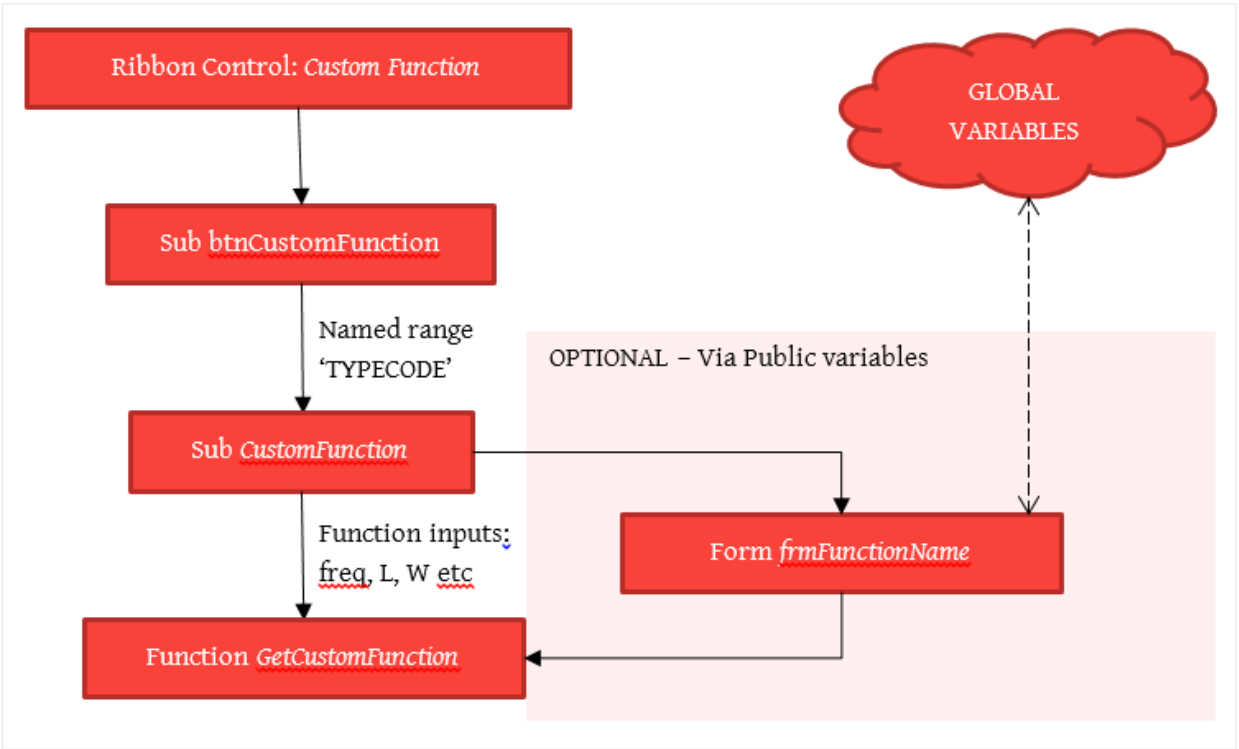

Figure 6: Program structure

In general, the sequence of events from a user clicking a button on the ribbon is as follows:

1. User clicks button on Ribbon – linked macro (*Sub btnCustomFunction()*) in *RibbonControl* module begins.
2. Macro in *RibbonControl* passes the named range 'SHEETTYPE' to *Sub CustomFunction().*
3. Sub checks for valid row range using *CheckRow().*

-----OPTIONAL FORM STAGE (steps 4 to 6)----------

4. *Sub CustomFunction()* calls the custom form *frmFunctionName.*
5. User inputs into fields in custom form *frmFunctionName.*
6. Code within custom form places data in public variables for use by *Sub CustomFunction().*

---

7. *Sub CustomFunction()* writes formula description.
8. *Sub CustomFunction()* merges or unmerged parameter column and applies validation.
9. *Sub CustomFunction()* writes formula in leftmost cell of template; including call of *GetCustomFunction().*
10. *Sub CustomFunction()* extends formula using function *ExtendFunction().*

3.6 Git / GitHub

Git is a tool that tracks the modifications to the source code and allows them to be integrated and developed. VBA, the programming language that sits behind the Excel, can be exported to a text file (.BAS file format) which is then scanned by Git.

GitHub is an online platform for collaboration on software projects. It is separate from Git, but communicates with the software easily. Contributors to the code will need to familiarise themselves with the following concepts:

- Branch
- Pull Request
- Stage / Commit changes

Briefly, the development process may proceed like this:

- An additional feature is requested by a user and discussed on the platform. The request would not require the user to have any working knowledge of the code.
- The proposed feature can be developed in a single environment (branch) by one or several contributors and discussed on GitHub.
- Once the feature is tested and approved, a Pull Request would be raised to merge the additional code into the new version.
- These changes are staged and committed to the code and rolled out to all users.

The exact steps for all future development will be formalised as the project continues to grow. In its current form, each contributor will be required to submit the .BAS files through GitHub.

It should be noted that a centralised version of the Add-In is editable by the user but is not able to be saved while in use by others. Should a change be made to the local copy of the Add-In, an analysis on Git will show the lines of code that have been changes from the original on GitHub.

4 FUTURE DEVELOPMENTS

It is hoped that future developments will come from a range of contributors across the industry. A preliminary rollout to a smaller group of testers may be considered in order have a reasonable level of Quality Assurance checking prior to full launch. A project site has been established for the plugin on GitHub and interested parties may sign up at any time. Contributions can also be made in the form of QA checking, preparing documentation or requesting features through the GitHub Platform.

5 CONCLUSION

Trace is a new open source Add-In to assist in performing acoustics calculations. The benefits of an Open Source development model have been discussed. Quality Assurance and error checking is easier with the source code available and viewable. It is hoped that development of the tool will be taken up by interested parties, either in a code-writing capacity, as well as for QA and documentation of the platform. Through development of *Trace*, the benefits to the wider acoustics community can be realised.

6 ACKNOWLEDGEMENTS

The author wishes to thank the entire WSP Acoustics Group, who have been willing 'guinea pigs' during the development of *Trace*. They have provided much useful feedback and support during the early stages. Such a project can only be nurtured in an environment where continual development and technical excellence are encouraged at all levels. This means spending time and resources on projects with long-term goals, not only for the company but also more broadly for the acoustics community.

REFERENCES

Bezroukov, Nikolai. 1999. "Open source software development as a special type of academic research: Critique of vulgar Raymondism." First Monday; Volume 4, Number 10 - 4 October 1999: University of Illinois at Chicago University Library.

dmahugh. 2006. *Custom UI Editor Tool.* 25 May. http://openxmldeveloper.org/blog/b/openxmldeveloper/archive/2006/05/26/customuieditor.aspx.

Druguid, John Seely Brown & Paul. 2001. "Knowledge and Organization: A Social-Practice Perspective." *Organization Science* 12 (2): 198-213.

Ghosh, Rishab A., Reudiger Glott, Bernhard Krieger, and Gregorio Robles. 2002. *Free/Libre and Open Source Software: Survey and Study Part V.* International Institute of Infonomics, University of Maastricht and Berlecon Research Gmbh.

Hars, A, Ou, S. 2001. "Working for free? Motivations of participating in Open Source Projects." *Proceedings of the Hawaii International Conference on Systems Sciences.*

Ifsttar. 2016. *I-Simpa.* http://i-simpa.ifsttar.fr/.

Madey, Greg, Vencent Freeh, and Renee Tynan. 2002. "The open source software development phenomenon: An anlysis based on social network theory." *Eighth Americas Conference on Information Systems* 1806-1813.

Microsoft. 2006. *Office 2010 Add-In: Icons Gallery.* 1 May. https://www.microsoft.com/en-au/download/details.aspx?id=21103.

Raymond, Eric S. 2007. *Goodbye, "free software"; hello, "open source".* Accessed July 9, 2017. http://www.catb.org/~esr/open-source.html.

—. 1999. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary.* O'Reilly Media.

APPENDIX A – CURRENT *TRACE* FUNCTIONS (VERSION 1.13)

Module	Description	Functions	Subs
Curve Functions	Rw, NR, NRC curves, as defined by the standards.	NRcurve NCcurve NR_Rate RwCurve RwRate CtrRate	PutNR PutNC PutRw
ImportFrom	Imports data into standard form	-	Import_FANTECH_DATA Import_INSUL
LoadSave	For importing blank calculation sheets or premade standard calculation sheets.	SheetExists RangeExists	NewTab SameType StandardCalc
LogFunctions	Universal logarithmic functions	SPLSUM SPLAV SPLMINUS	-
NoiseFunctions	Contains standard calculations and custom functions	AirAbsorb DuctAtten AWeightCorrections CWeightCorrections GetASHRAE GetFlexDuct GetERL GetRegenNoise GetRoomLoss GetElbowLoss freqStr2Num	Distance AirAbsorption Area RoomLoss DirRevSum *GROUP: MechElements* — DuctAttenuation — ASRAE_Duct — FlexDuct — ElbowLoss — RegenNoise — DuctSplit — ERLoss — Silencer
RibbonControls	Ports all ribbon control calls to the correct write macro	-	<NOT INCLUDED FOR BREVITY>
RowFunctions	Operations for calculation rows.	-	CheckRow ClearRw FlipSign A_weight_oct (legacy) MoveUp MoveDown SingleCorrection AutoSum TenLogN OneThirdsToOctave ExtendFunction ParameterMerge ParameterUnmerge
SheetFunctions	Inputs standard details into header block, finds project details, formats borders	-	HeaderBlock ClearHeaderBlock Update_ENGINEER GetProjectInfo ScanProjectInfoDirectory FormatBorders Plot HeatMap / GreenYellowRed FixReferences
VARS	Global variables	-	GetSettings