3. ALARM 1202, WE NEED A READ ON ALARM 1202

After I had completed my work and finished the *entirety* of the SPARGE™ programming, subsequently proving its accuracy and power by developing scale model 'micro-well' injection systems, and performing a myriad of performance tests of scale model air injection systems with the wells in both horizontal and vertical orientations (including *slant wells* at all angles of insertion), I found myself delivering lectures on the SPARGE™ technology around the Country [1]. These talks were generated somewhat 'by accident', as I found that the Industry's inquisitive minds truly wanted to know the Who, What, Where, etc. of the program and what it could do for their problem projects, many of which were languishing for *decades*. Without hesitation seemingly at each lecture, the same attendee question persisted, that being, "What took you so long?" Good question. Not that difficult to answer, but in truth the sheer *complexity* of the answer, and the work behind it, found most having difficulty grasping my response. This is because not only did one have to know the nuances of Finite Element Analysis and *how* Compressible Distributed Fluid Dynamics Systems worked, it necessitated their knowing of the progress and advances in desktop PC's, CPU's, Operating Systems, Random Access Memory chips, file storage devices, programming languages and all associated hardware.

When I began working on SPARGE™, the absolute very first action I took was to define a 'typical' medium complexity HAS remediation system and solve that system *by hand*. Literally, with a 500-sheet ream of paper on my desk, I laid out the Process Flow Diagram (PFD), developed *all* of the equations of state including tabulations of 'knowns' and 'unknowns', and started to solve the *entire* system foot by foot until I was finished and confident that my hand-calculations were sound and accurate. Now, here is the 'problem' with solving such a problem – by definition all distributed compressible gas injection (or extraction) systems are by the Laws of Physics mathematically *indeterminate*. This means that one cannot calculate the performance of *any* such system by linearly solving one equation, then using that result to solve another equation, and another, and so forth until the entire system's performance is accurately computed and known. This is because no matter how many equations of state one can derive, the sheer number of *independent variables* that one needs to compute the values to, to establish the performance of the 'system', far exceeds the number of equations that one *can* develop for that same system. In fact, for a 'typical' HAS system, there are approximately 33 *independent* variables that need to be computed (whose values are unique for every foot of the system), but *only* 17 *independent* equations of state that can be developed for that same system. One knows from simple 6th grade algebra class that to solve for two independent variables, one *needs* at least two separate and unique equations; for three independent variables, three equations, and so forth. So for 33 *independent* variables to be solved, one *needs* at least 33 separate and unique equations.

[1] Our desktop *Fish Tank Experiments*™, as we fondly refer to them, are available to prospective clients, Industry 'newcomers' and veteran Environmental Remediation practitioners alike. Requiring extensive attendee participation, the demonstrations provide an absolute wealth of extremely valuable *practical* knowledge to all, while dispelling the many myths and falsities about the effectiveness of *in-situ* air-based remediation employing horizontal wells of all sizes, lengths and orientation. Seeing is indeed believing.

Unfortunately fluid dynamics is not that forgiving, and hence this is the reason that the many who tried before me to solve the distributed gas injection system problem were stymied – one just cannot employ classical mathematical modeling/solving techniques to complete the job; the engineering and science behind the problem is just too complex. Brute force doesn't work, spreadsheets don't work, guessing doesn't work, nor does pleading, stomping one's feet or prayer – they all fail. The only way to solve such a complex problem is with modern computer technology coupled with a Finite Element Analysis approach.

I would be self-ingratiating and state that I drew such a 'wise conclusion' in a week's time (or so) after beginning my efforts, but if I did, I'd be stating a mistruth. I did not try the brute force or hope-beyond-hope approach in my early days of solving this problem. What I did do was try to develop additional compressible gas flow equations, which in truth I did, but their number fell far short of the number required – 17 became the 'magic number'. However, even falling short on equations, I *did* begin and eventually *did* solve the distributed compressible gas flow 'problem' *by hand*, and this effort took well over a month of continuous effort [2]! By doing so I developed an even greater appreciation for fluid dynamics and the complexities therein, as well as the experts, the savants in this field whose work I had studied extensively in college. They had no electronic devices of any kind to assist them in their efforts, just a pencil, paper, and maybe a slide rule or abacus. What they accomplished with so little to aid them was truly remarkable – Nobel Prize remarkable.

The second 'problem' with solving my distributed gas injection system problem, as I mentioned earlier, is the state of computational equipment i.e., PC computers, that were available to me *at the time*. To understand this 'problem', one is best to keep in mind that *at the time* the capabilities of personal computers and PC-based software were extremely limited compared to today's 'being taken for granted' PC's, calculators, tablets, and even common day cell phones. Quite like the Apollo 11 moon landing and their repetitive perplexing on-board computer's "Alarm 1202" display as they were descending to the Moon's surface, the capabilities of the Lunar Lander's *entire* on-board mainframe computer was *dwarfed* in capabilities and capacity by the then nearly completed commercially available HP-45 RPN pocket calculator. This statement is not a criticism, but rather a description of the state of computer capability at *that* time. The HP 45 calculator was the same trusty engineering necessity I kept alongside me throughout my collegiate years, and at which I remain amazed, even today, by its robustness and sheer beauty of engineering innovation. Putting everything in perspective, when I began my work on SPARGE™, anybody attempting to solve the distributed compressible gas 'problem' on a then-available desktop computer was *completely* stymied by the complexity of calculation, the need for large random access and hard drive storage space and programming language capabilities that were at their disposal *at the time* to complete this task. Unfortunately, unlike Apollo 11, I didn't have a Houston or Canaveral-based Mission Control and their collective team of engineers who could look up "Alarm 1202",

[2] In this statement, this means devoting an *entire* 8 to 10 hour day every day performing hand-completed paper and pencil calculations on my model 'system'.

establish exactly how impactful it was and suggest a workaround solution. I was left to my own Mission Control Engineering Team of exactly one individual – me.

One needs to recall that circa late 80's, early 90's, CPU's were primarily Intel model 8088 based, soon followed by 80286 and then 80386. The Pentium series of CPU's were just in their infancy. Operating systems (except for Apple OS 1.0) were limited to Windows 3.1, followed by Microsoft's Windows first Graphic User Interface (GUI) Operating System Win 95, and shortly thereafter Win 97. Due to a wide array of bugs in Win 97, Microsoft soon upgraded it to Win 98, this proved as 'buggy' in performance as its predecessor (but due to other issues, essentially 'created' by the upgrade itself). Finally, Microsoft delivered the first true near bug-free GUI OS that supported most 'heavy duty' scientific and engineering calculations on a PC by its Win 98 Second Edition OS (commonly referred to as Win 98 SE). But, with GUI's visual 'niceties' came a high price – up to 40% of *all* system resources were devoted to keeping the GUI interface visually attractive, completely functional and seamless. So, in conclusion, at the time that SPARGE™ was being engineered, the available PC's consisted of an Intel Series 80-series or Pentium CPU, Windows 97 or 98 SE, 256KB of RAM and limited space code and swap file saving hard drives.

On the scientific/engineering programming language side of the effort, matters were not that much brighter. One had the option of writing computer code in FORTRAN, which was my preferred choice *by far*, or try BASIC, C, or HTML. FORTRAN was the obvious best first option, however, once again, *at the time*, FORTRAN was a mainframe only programming language – one could write code but not develop a compiled program for use on a stand-alone PC, as doing so *required* the capabilities of a high level (considered at the time) mainframe computer to function.

To add insult to injury, to ensure that the code of the requisite complexity and encompassment *could* be computed correctly, one's PC had to be capable of meeting the rigors of the task, and this in itself proved in the end to be the greatest hindrance to completion. At the time that I was developing SPARGE™, PC's were expensive, very expensive. For a small business, investing $4,000 to $5,000 for the latest model PC, only to find that it was *not* capable of solving multiple dimensional FEA system calculations in either a *reasonable* amount of time or *accurately* [3] was a tremendous letdown (to say the least). This proved to be an extreme disappointment. One then had to wait for Intel or AMD to develop and place a newer, faster, more capable CPU on the market so one could swap the new 'brain' for the old. In doing so, it was common to find that the main board's CPU chip socket was of a different pin configuration (or needed many more pins that the older CPU), making the entire PC's motherboard unusable. Memory chips and SIMM boards yielded the same

[3] Because Sparge™ *requires* recurring repetitive multi-dimensional calculations to be done again and again until the system's *entire* performance 'solution' is derived, the necessity of employing the greatest number of *significant digits* (after the decimal point) in each calculation is crucial to complete the analysis *accurately*. Though limiting the number of digits from 16 to 4, for example, significantly speeds up computation time, it gives rise to calculation truncation 'error' that cumulatively builds up with each successive calculation.

frustrations. Win 98 SE was the first OS to permit the use of the then-new Universal Serial Bus (USB) thumb/flash drives, but these new storage devices were astoundingly expensive and highly unreliable. Finally there was the industry's then-current limitation on HDD capacity and swap file sizes. A 50 to 100 *megabyte* capacity hard drive was all that was available at the time, floppy disks were 1.2 megabytes and RW CD's were in their infancy.

Putting this all together meant that for some *years* the computational tail was wagging the dog – I had to wait for the computer industry to *create* PC technology, CPU's, OS and programming software that *would* be capable of handling the sheer computational complexity of the SPARGE™ FEA calculations, accurately and quickly. To place these needs in their proper perspective, it was quite common for a SPARGE™ performance analysis of a simple single-zone 250 foot long screened interval HAS well employing the most advanced 200 MHz Pentium PC that I could construct *at that time* to take several hours of computer time to complete. In stark contrast, our current SPARGE™-dedicated PC, employing the latest version of the program (this software version has more than a dozen times more requirements than any of its predecessors), analyzing a six-zoned ¼ mile(!) long screened interval HAS well *and* calculating all the heat transfer, thermodynamics and operational transients of the system *easily* completes its task in *less than* 10 seconds! The comparison is striking to say the least – in 'the early days' I *might* be able to complete 4 or 5 SPARGE™ 'runs' of a relatively 'easy' HAS system in an 8 to 10 hour workday. Now, I can complete *over* 50 highly complex *What if?* HAS system analyses [4] before lunch! And, with this increase in productivity comes an equal amount of computational precision (employing 16 significant digits in each calculation). History has proven time and time again that when constructed and operated according to the analytical requirements computed by SPARGE™, system performance and project success is 100% - there have never been any system failures or performance errors with any of the many hundreds of systems designed and/or analyzed by SPARGE™. The greatest deviation ever recorded between actual v computed system performance was that of the world's record ¼ mile long multi-zone HAS wells and their associated remedial systems that were designed and built for the Savannah River DOE site in Georgia. The deviation between calculated and actual system performance for each *entire system* was measured at an astoundingly small rate of .0015% *per foot* of well, this diminutive number falling just within the precision limits of the instruments completing the performance measurements!

[4] *What if?* analyses are one of THE most valuable capabilities of SPARGE™, as one can readily determine how any system will function if one or more key physical design or installation variables were to be changed. Such variables include placing the well higher or lower under the water table, changing slot sizes, propagations and % open area, revising applied injection pressure(s), overall screen length(s) and many, many more key variables.