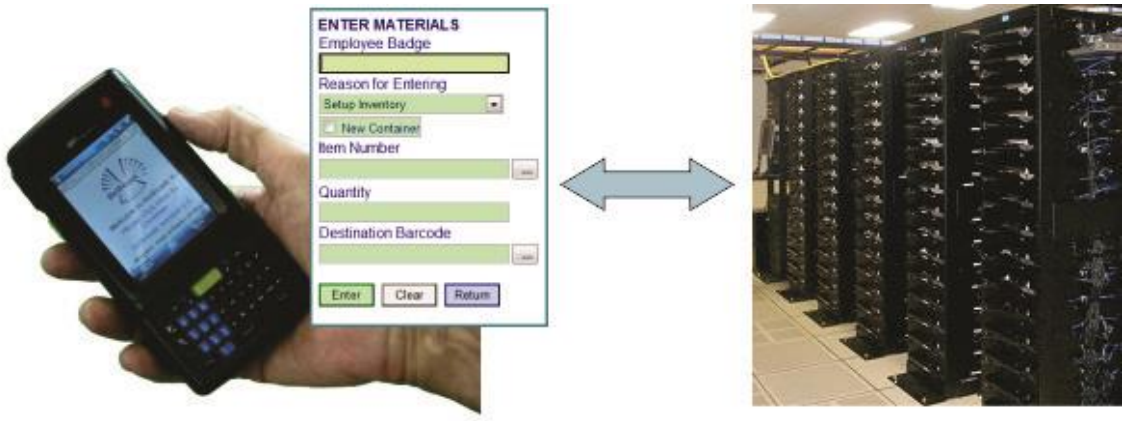
Failure to Excel in the Cloud

a cautionary tale by Dr. Peter Green



Introduction

I am a systems architect. I wrote this white paper as a cautionary tale that simply going from a client-server architecture to a Cloud-based architecture, with all computer processing being run at a remote data center, will probably not solve all of an organization's IT issues.

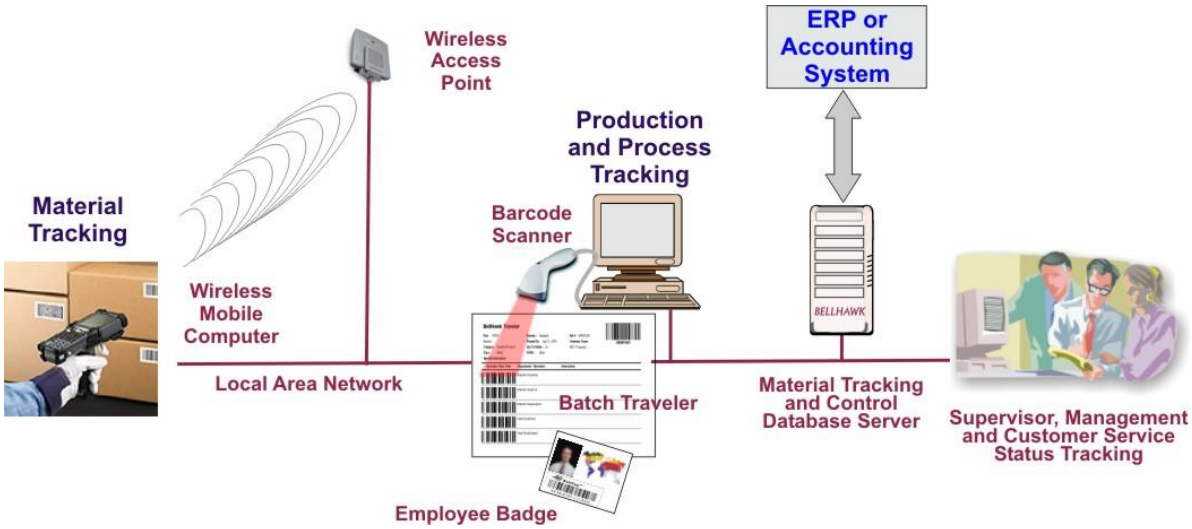
I have spent the past two decades working on barcode data collection systems for manufacturers and industrial distributors as technical director of a number of software and systems integration companies

My team started out in this AIDC (Automated Identification and Data Capture) field, by developing a simple single-user desktop application, to track operations in a PC board manufacturing plant. This application was written in Visual Basic, with an Access database, to replace an even-older "green screen" terminal application, which was tied to an old custom IBM mainframe application written in FORTRAN. This new application used a corded barcode scanner, which shared a "Y" connection with the PC's keyboard.

This single PC desktop system worked well, for about two days, until the production manager said "I love your new desktop production-tracking application but can you make it work so that all six of my work centers can each have their own PCs?

So, we rewrote the application as a client-server application with an Access/VBA front end running in each Windows PC and a SQL Server backend. This client-server version of the software worked well and we installed it in a significant number of manufacturing plants, a couple of which are still using this client-server software today, nearly two decades later.

This client-server application, a diagram for which is shown on the next page, worked well and was much-liked by many of our clients because it had a relatively simple database structure, from which it was easy to generate custom reports. Also, the front-end, being written in VBA (Visual Basic for Applications), was easy to customize.



But this system also had many disadvantages, which were:

1. The software could only be run on PCs connected to the LAN (local area network) within the plant or warehouse. Data could not be captured or viewed from outside the warehouse or manufacturing plant.

2. A separate body of code, written entirely in Visual Basic, was required to be loaded into each wireless mobile computer if mobile data collection, over the warehouse or plant's wireless LAN, was to be integrated into the system.

3. This mobile code, also required a local store-and-forward database in each mobile computer, so as to be able to reliably exchange data with the main SQL Server database. This was also needed to provide local point-of-action warnings when an operator or materials handler was about to make a mistake. This setup required a significant level of IT support as any change to the front-end code had to be installed on all the PCs and Mobile Computers used for data collection or reporting.

4. Unfortunately, the store-and-forward databases in the mobile computers could get out of sync with the main database, especially if the mobile computers were left where they could not communicate with the SQL Server database for several days. This could then require a substantial level of IT support to get the databases back in sync again.

Then along came the Cloud, with its promise of eliminating the need for any IT support at the local warehouse or plant level. Instead, all applications would run at a remote data center and users would interact with these applications over the Internet using a web-browser, with no need to load any applications software, into any end-point user device.

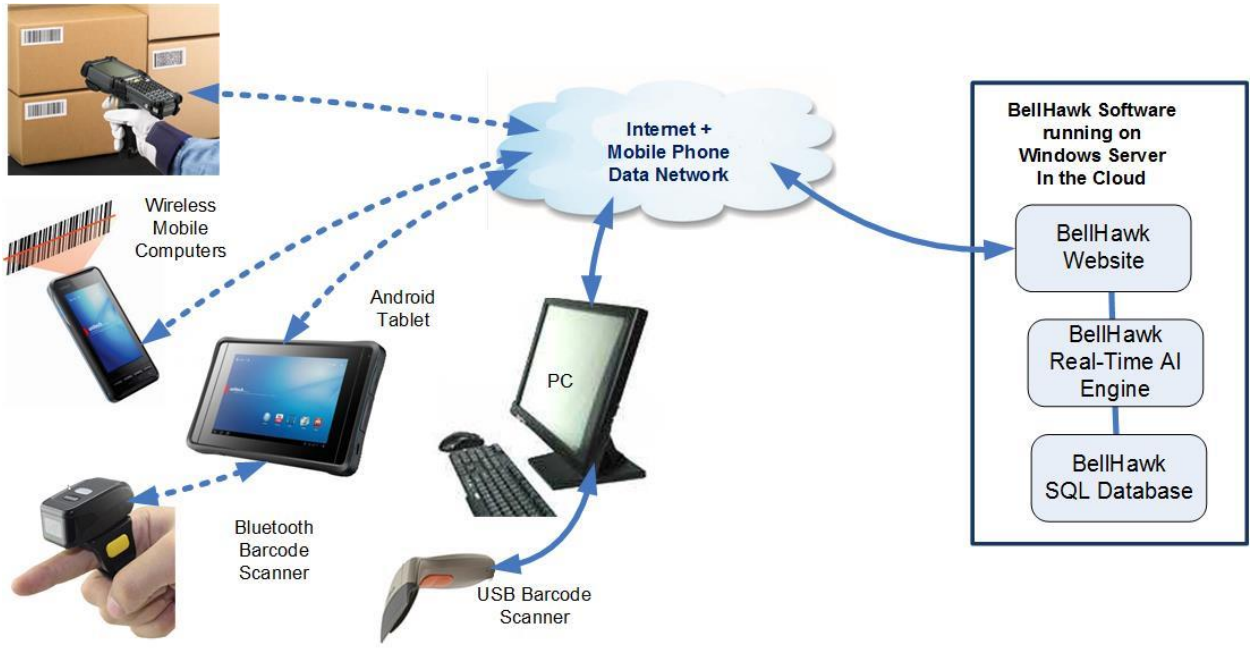
I would like to tell you that I saw this vision of Nirvana and immediately set to work developing a new Cloud-based version of what would become the new BellHawk software, to the sound of heraldic angels. But no. What actually happened was that Microsoft, to force software developers like me to conform to their new marketing vision, took away most of the features in VBA that we used to support our client server applications.

As a result, all applications, such as ours, had to move to the Cloud, as Microsoft steadily removed the compatibility features in Windows 7 and 8 which enabled old client-server VBA/Access applications still to be run. Then Microsoft drove a stake in the heart of these "vampire" applications with Windows 10, which no longer supported the needed compatibility features.

So, was this transition the path to the expected Nirvana? Well no. The use of the Cloud brought with it many of its expected advantages but also brought with it a whole new set of issues, which we are still working to overcome.

Pros and Cons of Cloud Applications

To put these comments in context, I will use the example of the Cloud-based BellHawk data-collection system, which is shown below:



This system, like its client-server predecessors uses barcode scanners, now plugged into PCs (now through their USB ports), as well as a variety of Android based mobile devices with integral barcode scanners or with Bluetooth connected barcode scanners, such as for hands-free applications.

The big difference is that we use a web-browser running in each of the end-point user devices to communicate with a website, written in ASP.net. This website then communicates through a rules-based expert system, written in .Net, to a SQL Server database.

This database, unlike the old client-server database is designed to handle many simultaneous transactional queries from a large number of data capture devices. It also checks each item of data, as it is scanned, selected, or typed in, against the database, so as to be able to provide immediate point-of-action warnings to users if they are about to make an operational or data collection database.

This works very well in that:

1. There is no need to load or maintain custom application code or databases on each end-user device. Also, data collection and viewing devices can access BellHawk anywhere there is an Internet connection. This makes deployment quick and easy and eliminates the need for any IT support for the data collection devices in each plant.

2. The BellHawk software is maintained in a remote data center, typically with backup power, daily database backups to another cloud computer, and a high level of physical security which is much better than having the server box under the IT manager's desk.

3. The BellHawk application software can be tailored to the needs of each organization by setting the parameters for the expert systems rules used by the AI engine without needing to customize the software.

We do give each client organization their own private SQL Server database and their own website instead of using a shared tenancy model. This enables us to ensure privacy and security of each client's data. It also enables clients to easily migrate their website and database from a shared server in the Cloud to their own server, if required for better performance.

This Cloud-based version of BellHawk is great for smaller manufacturers and industrial distributors. It has enabled organizations with no on-site IT staff to use barcode scanning to replace the use of paper forms and manual data entry into Excel spreadsheets in order to collect the data they need to run their operations. This automated data collection has also resulted in significant labor savings and reduction in operational and data collection mistakes.

The really huge advantage, however, is that data collection and viewing is no longer tied to the plant or warehouse LAN. Data can be collected and viewed anywhere there is an Internet connection, including over the mobile phone data network. This enables easy data collection and operations tracking in multiple locations. These locations can be off-site warehouses or they can be outdoors locations such as building sites and yards, which we were unable to handle with the older client-server technology.

But, as always, there is no free lunch. The big disadvantages that became immediately obvious were:

1. No local access to the BellHawk database, when running in the Cloud, at a data center thousands of miles away, making users unable to use tools like Excel, Access, and Crystal Reports to create custom reports from the database.

2. Even if clients installed BellHawk on a Windows Server in their local plant or warehouse, the database was now structured to handle transactional processing and not ease of reporting. As a result, only a few hardy souls were able to use things like SSRS (SQL Server Reporting Services) to generate their own custom reports directly from the BellHawk database.

3. Barcode Label printing. This requires a high bandwidth network link between the label data generation software, which gets its data from the SQL Server database, and each barcode label printer. This will not work at acceptable speeds over the Internet.

4. No easy way to exchange data with legacy, locally installed, accounting and ERP systems, as was relatively straight forward with the older client-server systems.
5. No easy way to tie a Cloud-based system such as BellHawk to process control equipment, weighing scales, and RFID sensors located in each plant or warehouse.

As a result of a lack of easy generation of custom reports, as was possible with the old client-server systems, we were pressured to provide an ever-increasing number of Excel exports from BellHawk. These exports enabled users to download the captured data in a familiar format and then to massage the data to produce the reports they needed.

Then we started getting reports of Excel exports from the Cloud-based version of BellHawk failing for clients who had large numbers of containers of material, and/or an especially large numbers of serialized items that needed to be tracked individually, in inventory. We also started seeing clients who wanted to import large amounts of data from an external system into BellHawk, in Excel, also reporting failures to import.

Hence the title of this paper "Failure to Excel in the Cloud" ☺

What was happening, on Excel report generation, was that the web-browser would make a request through the web-browser interface to fetch the data for the report from the database and create an Excel file on the Server, which would then be downloaded over the Internet to the browser. But, if a large amount of data was to be fetched, the web-browser interface to the server over the Internet would time-out before the Excel file was finished being created on the server. As a result, no data would be exported.

Unfortunately, unlike comma delimited files, these Excel files cannot be created and sent over the Internet incrementally. This is because Excel files have length and checksum data included into them for error checking. These cannot be inserted into the file header until the whole of the data has been inserted into the file and the file is ready for export.

A similar thing was happening on transferring Excel data into BellHawk, where an Excel file is uploaded to the server and then used to update the BellHawk database. The problem is that the connection from the web-browser can time out before the system has finished processing imported Excel file processing into new or updated database records. The timeout caused the web-server pool process, which was handling the import, to abort, which again results in a failure to transfer.

So how bad is this problem? We have found that our Cloud-based shared BellHawk transaction processors can easily handle Excel files with 2000 lines on import or export but beyond that failures may occur, depending on the complexities of the interaction with the SQL database.

For simple exports, not requiring recursive searching of the database, we have found that it is possible to get 10,000 records transferred reliably. But, in other cases, the limit may be lower, especially if the transactional processor is busy handling transactional data collection, which is given higher priority over Excel import or export. Or there may be overlapping Excel imports or exports from multiple users, which also can cause problems.

These limitations do not impede smaller manufacturers and distributors with a limited number of different parts in inventory but can be problematic for some clients that need, for example, to track large numbers of serialized items, such as electronic or pharmaceutical products.

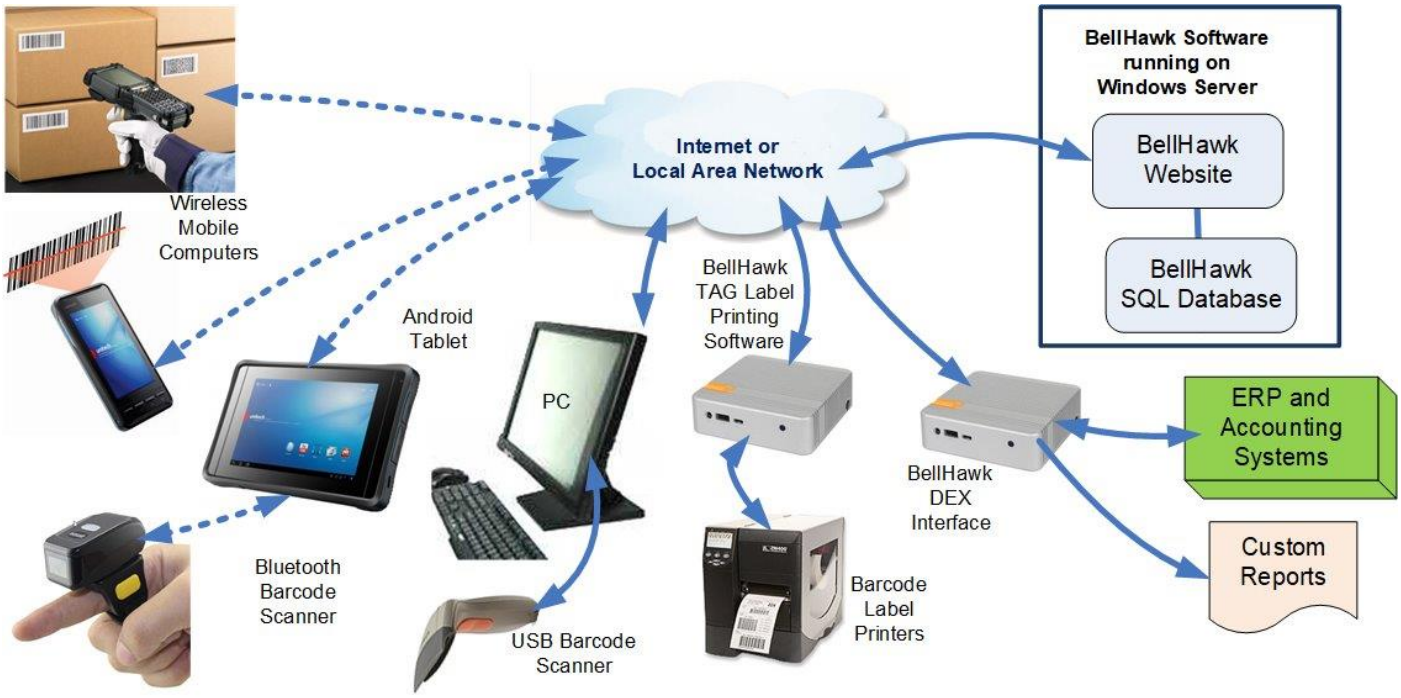
At a high level, the problem is that the BellHawk transaction processor is designed to handle a large number of quick interleaved small transactions from multiple devices. It is not designed to handle large volumes of data being exchanged with a single PC, which "hogs" the system.

This problem can be eased by running BellHawk on a more powerful dedicated Server with a separate processor and memory handling the SQL Server database from the processor handling the web-browser transactions. This will reduce the time to process SQL queries on large databases and therefore allow the import and export of larger Excel files. But this eliminates one of the major benefits of Cloud computing for smaller organizations, which is the cost savings from sharing the same computer with many other users.

Distributed Computing to the Rescue?

As we have seen, neither Client-Server nor Cloud computing is a panacea, especially for organizations with high Excel import and export volumes and other requirements, such as the need to exchange data with other systems and to produce custom reports.

My solution is to use distributed computing, using multiple networked computers:



Here we retain the benefits of the Cloud server to do transactional data capture but add in processing in other computers located where needed in plants or warehouses.

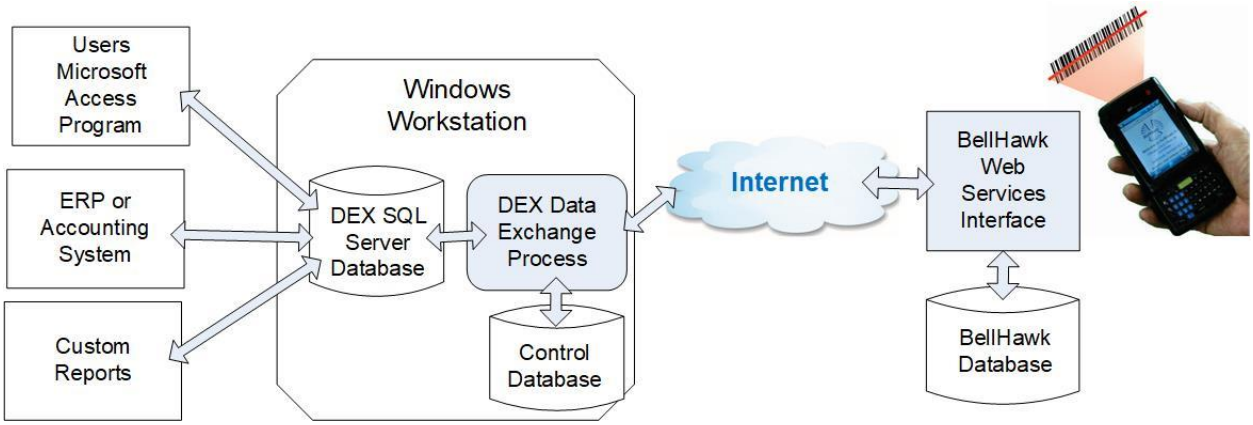
When a user wants to print a custom barcode label, they enter the request, typically by pressing a button on their device's transactional data entry screen. Then BellHawk uses a set of rules to

decide which label to print on which printer, and what data needs to appear on the label, and then places this small amount of information into the print queue in the Cloud server's database.

Software running in a Windows PC in a manufacturing plant or warehouse, long-polls BellHawk through its web interface and picks up the print request over the Internet, as soon as it is placed in the label print queue. This small amount of data is then expanded to become a large amount of data, within the PC, and sent out over the LAN to a local printer, which prints the barcode label.

In this way, we are able to work around the limitation of the slow speed of the Internet connection and its inherent unreliability, as well as not running into a timeout when transferring a large amount of data.

We do a similar thing with the DEX interface, which again runs on a Windows PC



Here, we use a program running in a local PC to exchange data, on a store and forward basis with a SQL Server Express database running on a local PC. Data written into tables in the DEX database are automatically relayed to the corresponding database tables in the BellHawk database in the Cloud. Also, transactional data captured by BellHawk is automatically relayed to tables in the DEX database where they can be used for reporting or data exchange with other systems.

This distributed processing approach brings back many of the advantages of a client-server system that we lost in moving to the Cloud. Users can link Access, Excel or a program such as Crystal Reports to the DEX database and use these to write custom reports. Clients can also use DEX to implement data exchange interfaces with their ERP and accounting systems. Variants on DEX are also available for communicating with weighing scales, process control lines, and RFID tag readers.

To facilitate these actions, the DEX database is structured like an Access database, which makes it great for generating custom reports or creating interfaces, while retaining the benefits of transactional data capture in the Cloud. Also, a version of DEX is available to be run on a Windows Server as a Windows Service to ensure data exchange continuity if a sever has to be rebooted.

With DEX, the transfers to and from BellHawk are processed through its web-services messaging interface, where each record is treated as a transaction, just like to or from any other user device. This enables DEX to ensure that large volumes of data are exchanged with the DEX

database reliably. It also ensures that each data object instance transferred is treated as a separate transaction, thus avoiding blocking users attempting to do rapid barcode scanning transactions.

But, as always, there is no free lunch. Data exchange between BellHawk and the DEX database is much slower than a simple Excel export made directly from the BellHawk database. This is because the Internet is relatively slow and DEX does extensive data checking to make sure only well formatted data is transferred. But once data is stored away in the DEX database, large Excel dumps can be made at high speed.

Also, large amounts of Excel data can be dumped at high speed into the DEX database by a user, to be transferred, once the user is done, to BellHawk at slower speeds with extensive data checking and without slowing data capture down.

With this change, we have, however, introduced "thick-client" computers back into each plant or warehouse. These computers need to be maintained by an IT person so we have diminished the value of moving to the cloud. Also, quite complex software needs to be loaded into these computers and setup properly before use.

The good news is that, with appropriate access privileges, all these computers can really be treated as a private Cloud, which can be maintained remotely, thereby eliminating the need for IT staff in each plant. Also, the software in the plant or warehouse level computers can be preloaded into Windows computers and shipped to the plant or warehouse, ready to plug into the LAN. This is especially true if these are industrial computers (such as shown above), running the Windows IIOT software, which enables automatic unattended restart when power is applied to the computer.

Recommendations

1. Start out using BellHawk on a shared server in the Cloud. This will enable you to try out BellHawk and see whether it meets your needs without needing to invest in a server or need any involvement from your IT staff.

2. If you are a small manufacturer or industrial distributor, with limited reporting or data exchange data exchange requirements, then this will probably meet your needs.

3. If you find that you are getting failures in Excel imports or exports or transactions are being executed too slow, then you probably need to run BellHawk on a dedicated (not virtual) server. This can be at a data center in the Cloud or run on a server in your own data center, or run on a Windows Server in your own plant or warehouse.

4. If you need to print custom labels on barcode label printers then you need to run the BellHawk barcode labeling software on a Windows PC in the local plant or warehouse

5. If you need to produce custom reports or interface with other systems in a plant or import or export large volumes of Excel data, then you should be using the DEX store and forward interface.

6. If you need to exchange data with other Cloud-based systems or to generate text message Alerts when events occur, then you should be using MilramX, www.MilramX.com, which is the Server based, "big-brother" of DEX.

Commentary

As we see from this example, "Moving to the Cloud" is not a panacea or an answer to all of our IT or operational problems, despite proclamations from a trade press to the contrary. Bear in mind that this is the same trade press which has also produced such technology insights as "AI will take over the world and kill us all".☺

Where we are heading is to implement systems, which run on networks of geographically distributed computers, linked by networks of varying speed and reliability. In an ideal world, we will place processing power where it is needed, and judiciously use expensive network bandwidth to optimize the cost-performance trade-off both in terms of people's time to develop and maintain these systems and their cost of acquisition and use.

This brings up the question of "What is Cloud computing?"

- Is it cost savings by sharing of an application running on one or more computers by multiple organizations?
- Is it saving on IT support costs by the remote management of one or more computers or applications by a team of people over the Internet?
- Or is it simply a different name for distributed networks of computers?

Distributed computing brings with it both the benefits and disadvantages of both Client-Server and Cloud Computing. Which is best, depends on the specific needs of each organization. The only thing that is clear is that there is no Silver Bullet solution, only organization-specific tradeoffs.

At the end of the day, the goal is to provide people with the information they need, when they need it, to do their jobs in the most efficient way. This may be in the form of reports or dashboards, text or Email messages, or even long Excel spreadsheets which can be analyzed at will. This will not be done with one computer or simple client-server or simple Cloud-based systems but probably by a distributed network of computers.

One final comment: Early in my career, we implemented an operations tracking system for a manufacturer of Microwave components. This early operations tracking system was written in FORTRAN and ran on an IBM minicomputer. When the application was up and running, we proudly walked into the company President's office and showed him the stack of reports, which we could produce each day with our new software.

The President immediately brushed these aside and said "What I want your computer to do is to show me the six things that I should be worrying about today; everyday". And with that, he simply walked out of the meeting.

Half a century later, we have not achieved his goal. We have come a lot closer, especially with the application of AI methods. But, at the end of the day, we have not yet achieved this goal.

Will we finally achieve this goal through the use of AI and distributed computer systems, or do we need the next big leap forward, such as Quantum Computing?

Author

The author of this white paper is Dr. Peter Green who is the Technical Director for KnarrTek. He earned a BSEE and a Ph.D. in Computer Science from Leeds University in England. He was a Senior Member of the Research Staff at MIT and a Professor of Computer Engineering at WPI.

Dr Green has been implementing real-time operations tracking and management solutions for many decades. Dr. Green is a domain expert in automated data collection as well as in materials tracking and traceability. He is also an expert in Real-Time Artificial Intelligence and its application to solve complex operations management problems. Teams led by Dr. Green have implemented over 100 inventory tracking and operations management systems for clients over the past decade, including manufacturers, industrial distributors, food and pharmaceutical processors, medical and biotechnology laboratories, as well as repair and service organizations.

