Oracle recently changed their strategy for their GRC software and is moving towards cloud-based solutions. My sense, from hearing from friends and colleagues in the space, is that Oracle's change in strategy is confusing to its risk advisory and implementation partners. If these practitioners are confused, I can only assume that customers are confused as well.

I wrote a LinkedIn post about this and heard from various customers and practitioners that confirmed this frustration. Based on that, I feel the need to discuss the specifics of the GRC software strategy change, but also put this change in context by looking at the bigger picture as well.

First, let me address this specific change in GRC software strategy. In order to do so, I need to provide some background to how we got here. Dating back 10 years there were many more organizations that were attempting to build software solutions to address what I have referred to as the controls automation space. The features that differentiate these software companies were:

1. Software built specifically for Oracle E-Business suite
2. Development required an understanding of Oracle's EBS architecture and what made it unique from other applications
3. The goal of the software was to provide automation such as:
    a. Segregation of Duties and single function risk monitoring and preventive controls
    b. Automation of user provisioning
    c. Automation of quarterly access reviews
    d. Automation of month end close
    e. Monitoring of activity in the applications where an audit trail doesn't exist – for example - elements subject to change control, supplier master changes, and security changes.
    f. Expedite development of user interface changes

This type of software wasn't just generic in nature. To be effective, it required specific knowledge of the Oracle EBS applications and related architecture. For example, if you had an SAP mentality and applied that approach to building controls automation software for Oracle EBS, you were lacking an understanding of key differentiators between SAP and EBS. Companies that provided controls automation software in this space included Logical Apps, Applimation, Approva, CaoSys, Absolute Technologies, Phulaxis, Greenlight Technologies, and Oracle. Oracle's solution, called Internal Controls Manager, was 'middle of the road' in the space, at best.

In 2007, Oracle made a major acquisition of its largest competitor, Logical Apps. Logical Apps had recently acquired Applimation's GRC software line. At the time, unofficially, I ranked Logical Apps and #1, Applimation as #2, and Oracle towards the bottom in terms of features and effectivity. However, with the acquisition of Logical Apps, Oracle purchased its #1 and #2 competitors.

At the time of the acquisition, Applimation's and Logical Apps' software were both embedded as a custom schema in the existing EBS technology stack. In essence, it didn't require additional hardware. Subsequent to the acquisition of Logical Apps (and Applimation's GRC code via Logical Apps), Oracle decided to take a different approach to the market. Rather than embedding the software within the Oracle EBS tech stack, they decided to build cross-platform software that would work across applications. Thus, they re-architected the applications to be built on software that would be independent of the EBS tech stack.

This change may have made sense to reach large enterprises that used various ERP systems to run their business, but it proved to be a horrible move for the SMB market. The change significantly increased the complexity and cost of the project just based on the additional hardware and tech stack changes that would need to be added to the scope of the project.

Additionally, Oracle significantly increased the price of the software that was being offered by Logical Apps and Applimation. Having taken out the #1 and #2 largest competitors, it could afford to do so with less competition – so at least Oracle thought… What this did was effectively price out most organizations in the SMB space and made it difficult to show ROI for the project even for large organizations.

There have been a lot of other 'moves' in this space by other providers since then that are outside the scope of this article. Around this time Oracle purchased Logical Apps, I had been an advisor to various of these companies – providing analyst services related to product direction and content. Because I wasn't with one of the big analyst firms and provided direction to Oracle's competition, eventually I was shunned by Oracle in its product direction. However, I had and continue to have various channels that continued to provide me feedback on Oracle's strategic direction. Out of this period, I have primarily aligned myself with CaoSys which has emerged as the most significant competitor to Oracle.

For full disclosure, my firm is a reseller, provides implementation services, and licenses content to CaoSys for three of its solutions – CS*Comply, CS*Proviso, and CS*Audit. In my biased opinion, CaoSys has emerged as the most significant competitor to Oracle.

During this period, I heard from various organizations that were struggling with this change in architecture. Two issues that become a challenge early on were the performance of the conflict scanning process and the use of preventive controls as part of the provisioning process. These eventually were worked out by Oracle, but took time.

During the past few years, Oracle has also re-architected the tech stack for these solutions in an attempt to consolidate the number of servers needed.

All told, from a customer's perspective, Oracle's change in technical architecture from being embedded in the Oracle EBS tech stack to being outside the EBS environment altogether has been a challenge for many customers and has significantly priced-out the cost of the project for most in the SMB space and also a percentage of large enterprises.

If we could go back to the day that Oracle purchased Logical Apps, I would strongly recommend to Oracle to leave those applications in the tech stack and to have built another application to tie the 'best of breed' GRC apps within each of their solutions (EBS, PSFT, JDE, Fusion, Siebel, etc). The little value this cross-platform capability has provided to a small number of clients has been greatly overshadowed by the challenges to the far greater majority of organizations that have implemented or could have implemented the applications if they have used the existing technical architecture.

## The Latest Changes

This leads me to the latest changes. Oracle, as a whole, is pushing a move to the 'cloud' for its applications. With this change in overall strategy, Oracle is pushing to move their GRC applications into the cloud as well. This is another significant strategy change in its architecture of the applications, similar to the change it undertook when it transitioned the Logical Apps (and Applimation) solutions

from being embedded in the EBS tech stack to being on its own hardware. This strategy may be beneficial as it relates to reducing the need for additional hardware for these applications (since they would be 'hosted' in the cloud), but it is the wrong direction.

Oracle GRC applications require deep and close integration with the applications tech stack. Many of the benefits of the controls automation aspects of the GRC suite come from enabling technology within the technology of the applications.

Let's take one example – Oracle's CCG solution. CCG has two components – one that is based on snapshot technologies and one that is based on the use of triggers. The solution based on triggers creates an audit history for DML activity (Inserts, Updates, and Deletes) that happen in the database. Primarily the solution monitors changes made by users logged into the EBS application, but these triggers also work when an employee is logged into the database directly. Oracle provides various seeded content to monitor commonly used processes within EBS. Oracle also provides a mechanism for developing custom content.

In theory, to enable the 'seeded' content provided by Oracle and/or extend the solution with custom auditing, the cloud application will need to be architected to enable and/or insert various new triggers within the database. This flies in the face of what you'd think a 'cloud' application would do. The architecture of this solution is that you need to provide a solution hosted in the 'cloud' permissions, through your firewall, to enable and create triggers within the database.

Another example of this level of integration is the automation of controls within the provisioning process. In simple terms, when a security administrator is attempting to assign a new Role or Responsibility to a User, a trigger suspends the transaction and analyzes the Responsibility assignment to see if it creates a risk. If a Segregation of Duty rule is enabled (or multiple SoD rules are enabled), the provisioning process within EBS will be suspended until this information is passed to the 'cloud' solution for it to be able to analyze the current access for the User along with the desired access (i.e. new Responsibility (ies)). The cloud solution will need to analyze this data and send the result back from the cloud through the firewall to EBS, which is waiting until it receives its response from the cloud instance. When Oracle re-architected this solution WITHIN the same data center, it had serious performance issues. Now add to this the technical challenges with interfacing to an application hosted in the 'cloud'.

There are many other examples (PCG preventive controls come to mind) I could provide, but I think you get the point.

And as of the writing of this version of the article, Oracle is pushing the 'cloud' solution, but has not yet developed the 'local agents' necessary to ensure that the preventive controls will even be supported. Similar to what it did when it re-architected the Logical Apps solutions, Oracle's overall strategy is being forced on a solution that really is unique in its product suite.

## The Big, Big Picture – Oracle needs a GRC Evangelist!

This brings up another long-term big picture thought that I have had for a long time. Why isn't there anyone at Oracle looking at the big picture related to GRC issues? When I say GRC issues, I don't just mean the GRC software, but everything that happens within the suite. This is just another of the many missteps, in my opinion, that Oracle has made. Let me illustrate some big picture GRC thoughts.

1. Everything CCG does should be built into the core applications. This includes both the change tracking components (DML – Inserts, Updates, Deletess) as well as the snapshot features – both have value.
2. Oracle allows SQL injection in nearly 60 forms with no monitoring as to the changes.
3. Similar to SAP, Oracle needs to provide a mechanism to move changes (objects and data) from DEV to TEST to PROD. SAP does this through the transport mechanism.
4. The preventive controls during provisioning need to be a part of the core applications. I think it would be difficult to standardize the SoD content that is needed, but 'engine' of the approval and preventive controls should be a part of core applications.
5. A GRC evangelist needs to be integrated with the software requirements within the Oracle development cycle. One example of a poor decision by development is documented in my article "Manage Proxies: Pandora's Box Wide Open" which can be downloaded at www.erpra.net. The ATG development team released some 'enhancements' to the Manage Proxies feature that left this capability wide open. In reality, if I were the GRC Evangelist at Oracle this 'feature' would never have been developed.
6. There are dozens of other changes I'd suggest making within the core functionality of the applications – enhancements to the journal approval workflow, shutting down the ability to do multiple adjustments against the same transaction in AR, significantly limiting the profile options that can be set through the Personal Profile Values form, significant enhancements to the Value Set Security feature rolled out in 12.2 (and backporting of this feature to 12.1) – to name just a few…
7. User Interface changes – Oracle needs to STOP re-developing everything in OA framework. The design of certain forms in OAF is a nightmare – customer master and supplier master would be at the top of this list.
8. Oracle needs to produce a matrix of what forms/functions are local versus global. This is key to understanding security and change control. For example, within the PO module, Line Types are global (i.e. apply to all Operating Units). Document types are local (individually set for each Operating Unit. AR Customer Profile Classes are global. AR Transaction Types are local. Not having this documentation is a nightmare for security and controls design.
9. Best Practices documentation is not updated for key changes and not maintained on a timely basis.
10. Best Practices for monitoring of DDL changes within the database needs to be developed (currently the best documentation on this is provided by Integrigy).
11. Code migration and code management should be a part of the core applications, not a third party tool.
12. There is need for much more encryption throughout the applications to protect sensitive data.
13. Oracle should NEVER have provided the ability to de-crypt credit card and bank account data. These programs need to be removed from the applications altogether and only provided by Oracle Support with a signed agreement acknowledging the risk of using such.

So I started this article by stating the Oracle's GRC software strategy is flawed. In reality, their overall strategy for the applications are flawed. I recently did a training class for the PCAOB. At the beginning of the class I stated that Oracle is nearly "un-auditable." When the class was over one of the attendees said they could see where I was coming from with that comment. Perhaps now you realize why Oracle's

GRC strategy is not only flawed, but they need a senior-level GRC Evangelist to come in and right the ship.