

## Overview

Access was introduced by Microsoft in the early 1990's as their database management tool intended for end users and as such made it part of their Office suite of applications. Since then, it has been upgraded to conform better with today's paradigm of object based technology. However, it still lacks the sophistication of current development tools such as .Net and Visual FoxPro. The author's experiences in working with Access has exposed several weaknesses that must be considered when developing applications using this tool. The key issues are outlined here in order to expose stakeholders to consequences and methods of mitigation when developing applications with Access.

Unique to Access is that all application components (forms, queries, reports, etc.) can be, and usually are, stored in a single ".MDB" file. This is a blessing for end users who do their own development but a curse for professional. Many of the weaknesses can be attributed to this as we will see in the following paragraphs.

## Productivity

Access and Visual Basic are object based but not object-oriented (OOP). True object-oriented programming tools have three primary requirements, inheritance, encapsulation, and polymorphism. This essential eliminates reuse of code and frameworks requiring hours of tedious work to reinvent the wheel. This is especially problematic when it comes to error trapping as there is no global error handler with Access. Error trapping must be accomplished by brute force in all methods (subroutines). This can be mitigated to some extent by implementing a function to display the error information and accept user input, but the error must still be trapped, the function called, and the response applied to resolve the error. This requires considerable time to code, test, and debug. As a consequence, many professionals and most end user developers will choose not include error trapping in applications.

There are a number of native visual controls that the author believes are lacking in Access and thus depends on ActiveX components to fill this gap. The Access report engine is sufficient for basic applications but falls short when more complex reporting is required.

## Maintainability

The single "MDB" works well for a user developed single-user application. But once you cross over to the multi-user realm, maintenance can be daunting. For example, you have created a single "MDB" Access application and it has been in use for several weeks or even months and is now populated with large amounts of data. Now several changes are desired. If you are the only user, no problem. But if you have several users, they will have to wait while you implement the changes. Or even worse, you have to send the "MDB" file to someone else to do the work; now everyone is at a standstill! A good solution is to use two "MDB" files, one for data and the other for everything else. (The author uses a third as a "library" for common functions.) Since Access 2003, a new feature "Database splitter" which will take a "MDB" and split out the tables to a new "MDB" file that can be located in a shared folder. This will allow work to be done on the application "MDB" separately from the data and the it can be distributed to the users.

## Reliability

Access has a propensity for file corruption. Back up regularly and often. Be sure all users have exited the application, as backup requires exclusive use of the database.

## Performance

There are a lot of factors which affect performance. Paramount is good database and application design. But for purposes of this discussion, the assumption is that a well designed system has been implemented.

So how does Access stack up to FoxPro and .NET? Numerous tests were designed in two areas, object instantiation and data manipulation. Data manipulation test were done in two modes. Local mode where the data is located on the same computer where the application runs and remote mode where the data is located on a different computer (or server) and accessed over a 100MB network. As .NET is dependent on external database engines (such as Microsoft SQL, MySQL, or PostgreSQL) only remote mode test were done with .NET. Additionally, the data test were repeated under escalating stress representing 5, 10, and 25 simultaneous users. Instantiation tests included a form with approximately 50 visual object, a form with 10 ActiveX visual objects, and various instantiation of non-visual object (VFP and .NET only as Access and Visual Basic are not OOP).

The author was not surprised that Visual FoxPro was the top performer in all the tests and by orders of magnitude in some of tests. Access was a somewhat better at instantiation than .NET which was not a surprise as the .NET runtime is huge. Local data manipulation by Access was better as long as the stress level was low. For remote data, Access was on par with .NET for less than 5 users but quickly deteriorated as the stress level was increased.

## Security

Integrated with Access are two security methods. First is to password protect the "MDB" file. The other is a permissions based user-level security. These security features will prevent unauthorized access when using Access or an Access application. However, there are still ways to open the "MDB" file and view the data. To prevent this, Encryption is required and Access does not have this ability. Remember that locks and keys only keep honest people out.

## Conclusion

Access provides an entry level database application development tool that can fill the needs of many end users who desire to develop on their own. This is certainly the case under the following conditions:

- Performance is not an issue with at most a handful of simultaneous users.
- Security is of low priority.
- The level of productivity and maintainability are not critical as the user/developer has lots of time to devote to the project.

Experienced professional developers tend to avoid using Access primarily because of productivity and maintainability issues described above. Greater productivity means reduced development time, superior products, lower costs, and happier clients.