White Paper

embarcadero

Mobilizing Enterprise Data

Top 5 mistakes and how to avoid them!

By Stephen Ball, EMEA Technical Sales Consultant (RAD)

& Associate Product Manager (InterBase)

Embarcadero Technologies

March 2014

Revised May 2015

Table of Contents

EXECUTIVE SUMMARY

Many businesses today are looking to identify opportunities to increase efficiency in their business processes and to expand opportunity by using apps on mobile devices. While this is positive, organizations must manage the initiative correctly in order to minimize risk of failure, as getting it wrong can cost much more than just the initial budget allocated for the project; in both time and money, it can severely damage an organization's reputation.

Mobile is well suited to new working practices from a technology standpoint, but there are a number of issues that organizations must consider from legal and practical perspectives to ensure the best outcome in the longer term. These include:

- Data governance to enterprise architecture
- Data protection compliance
- Business risk with off-site data
- Storage and management of at-rest data on device
- Operation change / reliance risk

This report summaries the challenges of mobilizing data for business and articulates the most common mistakes that organizations make, alongside providing guiding principles for mobilizing enterprise data. The report is intended for CIOs, data controllers, custodians, owners, stewards, database administrators, software developers as well as legal departments within an organization.

SITUATIONAL ANALYSIS: BACKGROUND AND LESSONS IDENTIFIED

Adoption patterns show that it takes around 10 years for a technology to develop and gain wide acceptance and use. Looking at the progress of the Apples iOS and Google Android platforms, and touch-based smart phone and tablet technology, this view appears to hold true. It is seven years since the first modern "touch" devices were launched. Already, modern mobile devices have consolidated core feature sets around touch, geo-location services, cameras and fast 3G, 4G access; and powered by the operating systems, made the mobile phone an instant-on, mini-sized pocket computer.

The earlier generations of mass-use mobile devices, driven by then market leader RIM (Blackberry), delivered tremendous capabilities around communication and enhanced email. Businesses were fast to exploit the technology for staff productivity gains. It should be noted that this early platform

was enabled by the availability of data via mobile networks, but the push for using these devices came from the business world.

Today however, the mobile phone market has matured and it is consumers (not businesses) that are now driving demand for these devices. Current platforms deliver against this demand by effectively packaging traditional business features like email with social tools such as Facebook, Twitter, LinkedIn and others. A large eco system of visually rich, low latency applications that are optimized for instant-access anywhere is further supporting this trend.

Smart phones now dominate market share that provides business with a pre-tooled workforce that is already trained on how to use these powerful devices. This has led to wide acceptance of Bring Your Own Device (BYOD) and Choose Your Own Device (CYOD) as tooling strategies helping to manage the cost of enterprise mobile hardware.

The current device manufacturers have learnt lessons from Blackberry's experience, especially from a technical point of view. Amongst the most obvious ones are application speed powered by on-device data and the ability to view and respond to the data offline. Offering these capabilities on devices today has greatly enhanced customer satisfaction and usage. These capabilities have in turned filtered into the development of features in a large number of mainstream applications

Interestingly, Facebook was originally developed as an HTML5 app. When this was redeveloped to be a true native app the result was users then spent double the time on Facebook overnight. Why? Native code base apps provide a much richer user experience, partly powered by direct access to local data caching. Consequently, more advertisements began to be viewed on Facebook causing an increase in its share value by 18%!

This indicates the value of true native apps and that cached data increases user acceptance and productivity. However, for many enterprises local data storage creates challenges around data security and safe access on mobile.

The modern workforce is use to undertaking data entry and analysis on mobile devices. This is relevant for both "customer facing" and "internal facing" requirements. Increasingly, businesses are taking their customer interaction to mobile platforms and are looking at how they can power their work force to achieve productivity gains for their organization in an effort to mirror and surpass the gains that email originally delivered to the business.

TOP 5 MISTAKES WHEN MOBILIZING ENTERPRISE DATA

There are a number of mistakes that enterprises commonly make when undertaking initiatives to take data to a mobile environment.

MISTAKE 1: TRYING TO MOVE "EVERYTHING" TO MOBILE

The purpose of a mobilization project should be to provide specific functionality that addresses the needs of the end user when mobile. Answering the question - "Is this functionality necessary for the mobile app to achieve its goal?" – at the requirement assessment stage is imperative, but often overlooked.

A clear understanding of which features need to be available in the mobile app and which are not central to mobile success is paramount to defining the scope of the project, along with defining the responsibilities that each application will have within the overall business architecture. For example, while data entry and analysis may be performed using core enterprise software, if data capture can be supplemented using mobile to reduce workload then the function merits being available on mobile. The data analysis can still be maintained and run from the core enterprise software.

Similarly, as mobile apps typically only cover a subset of the core enterprise activity, they therefore require only a subsection of the enterprise data. Determining the operations that will need to be performed on mobile will help define the mobile data requirement. Only the relevant data must be stored on the mobile device.

Mobile data costs money! When devices are not connected via wire or WiFi, they typically incur data costs. There is a cost associated with storing data on the mobile; and a hidden cost of moving excess, unused data because it increases the load on servers, devices and network traffic, all of which can be avoided. This is true for both pushing data to devices and also pulling data back from devices. If data storage is inefficient at logging changes then data processing increases both ways. The tracking of changes should be managed in the data layer rather than in the app as this removes the chance of inadvertent omissions.

HOW TO AVOID THIS MISTAKE?

Be clear up front! Data modeling is a great way to get clear visibility of the logical model and the breadth of the data required to complete a mobile project. This also visually improves communication and risk/impact analysis associated with the data and facilitates clear business context for data and its use by removing data obscurity.

Beyond simple modeling, collaborative team tools can further help achieve these goals as they bring data structures to life and enable developers, data controllers, database architects and data consumers to clearly understand the use-case for the data that is proposed for mobilization.

MISTAKE 2: NOT MAINTAINING COMMON ENTERPRISE METADATA

Metadata defines field sizes, data types, field names and the purpose of the data in question. It is fundamental to data governance best practice. Using the same metadata across the enterprise ensures that data is compatible from its source through to its destination.

When organizations develop applications, it is easy to fall into the trap of re-inventing the wheel with regard to data structures, which in turn introduces incompatibilities between data sources. This is not an issue for mobile and central databases only, but also for the same fields used in multiple tables in the same database. Envisage a situation where the 'Person Name' field in one table stores 50 characters and in another 30 characters. When the data is merged, there is a potential of data integrity issue leading to data loss if the name is 31 or more characters long.

Metadata also encompasses the business rationale for the records or information. Organizations must be able to justify the business reason for storing data and ensuring that it adheres to the guidelines outlined under the Data Protection Act. Metadata needs to be 'live' to be useful. It's no good being a book gathering dust on the shelf.

HOW TO AVOID THIS MISTAKE?

Having a defined, accessible metadata dictionary is key to avoiding data incompatibilities. By using data domains, organizations can define a specific

field type (e.g. Name = 50 characters) so that any change to the domain is automatically updated across other databases.

Domains must be created in logical models and used to define fields in physical databases. Using InterBase (a multi-device, embeddable and scalable relational database) supports domains that also include check constraints so the business logic around accepted values for that domain is consistently enforced.

The data dictionary must be visible, accessible and subject to query by the entire team. Clear business knowledge of the purpose of the fields and the value types to be stored must be managed in the data dictionary. Rather than just storing this in a document, tools like Embarcadero ER/Studio Team Server can help bring a data dictionary to life and also enable rich contextual information to be displayed to developers, data consumers, architects, controllers etc though an online portal and also via a REST API that can be integrated into other core systems.

This all speeds up development and deployment for mobilization projects – it removes guess-work, bottlenecks and mistakes around how the data works by ensuring it has a coherent and visible architecture that is maintainable in the long run.

MISTAKE 3: AVOIDING LOCAL DATA STORAGE

The key reason to mobilize a workforce is to provide productivity gains and to enable employees when in the field. While enterprises have security concerns around storage of data on devices locally, offline data storage is essential for a number of business reasons.

From a practical standpoint, there are very few tasks that require live connections - for example, a reservation system may require a live data feed to check for resource availability. For most other scenarios, offline data processing capabilities bring benefits. Offline processing is often faster as data is not being pushed and pulled constantly (which also reduces data costs), but most importantly access to the data is always available. To illustrate this; a sales executive arrives at a customer site to take an order, but is unable to secure a mobile signal to connect the device. If the executive's objective for that meeting was to take the next order based on previous usage, instantly, the executive's productivity is reduced, as the goal of the meeting is not achievable. It this illustration, additional work on the part of the executive after the customer visit, is required to process the order. With local data stored on the mobile device, the sales executive would be able to complete the task at the

client site, despite the lack of connectivity. Email is the classic example of why offline storage is of benefit to usability.

HOW TO AVOID THIS MISTAKE?

The key to a good data cache is keeping it updated. Identifying data deltas (i.e. only the changes made) for transport to and from local data caches is essential to facilitate the best user experience with the least moved data – the promise that a data cache aims to deliver. However this is not trivially achieved!

Until recently, two common approaches have been used by developers and replication engines to enable identification of changes to data.

- Logging changes as they happen using table triggers and a log table
- Tracking when data changes using extra time stamp fields

While suitable for server side data movement, these approaches can struggle with mobile environments.

Log tables become verbose quickly as every record change is logged. If one record is updated 10 times, there will be 10 entries in the log. Adding field level changes increases the number of logged changes each time by the number of fields changed.

Tracking with time stamp fields introduces different issues e.g. controlling concurrent updates: Table locking is often required to ensuring changes made a kept visible to the right destinations, reducing the scalability of this approach. Field level tracking also requires additional fields for each tracked field, slowing down the update process.

Neither approach is ideal for field level change tracking and suffer specific issues when scaling to large numbers. – InterBase Change Views provide a new approach that is suited for mobilizing data to multiple destinations and is safe to have multiple users updating together. Using a subscription-based model that tracks at field level what has changed, it is highly scalable for both server side and client side data tracking. Using Change Views can drastically reduce the amount of data that is required to be moved, enabling developers to deliver the smallest possible deltas change tracked uniquely for multiple destinations.

MISTAKE 4: SETTLING FOR DATA STORAGE ON MOBILE THAT DOESN'T SUPPORT GOOD PRACTICE IN DATA GOVERNANCE OR DEVELOPMENT

Enterprise software architecture in the past has typically comprised of server and desktop systems making data storage relatively simple to manage as far as

(typically) having one major end-user platform to contend with. Mobilization projects are multiple platforms by nature, leading to a number of challenges around ensuring that not only each target platform, but in the case of Android - each vendors implementation of that platform provide comparable capabilities. This is very important if your architecture relies on data storage or encryption capabilities that the client platform provides.

An objective of good practice around data governance and software development (e.g. practices like writing PIA's (Personal Impact Assessments) or following broader practices like ITIL (Information Technology Infrastructure Library)) includes confronting and managing risk through the life cycle of the application by identifying risks upfront or during continuous service improvements that are undertaken.

To achieve the balance between good practice (when working with a diverse range of devices) matched with the needs of the organization to deliver software in a timely manner, the development process benefits from having cross platform database support through the development lifecycle. Selecting a database that doesn't provide the same feature capabilities to the development team on their chosen development machines compared to the target of the devices deployed to will introduce increased project risk; time needed for development, testing, deployment and change requests will also increase and be higher risk in nature. With the popularity of BYOD and CYOD the diversity that development and testing teams have to deal with is vast and growing, which compounds the effect of settling for a database that doesn't support multiple platforms in a unified way.

The 'data controller' role should be central to the data usage policy and implementation / enforcement of it. Data visibility really should be part of the data layer as it is an integral part of the data security. It is often a mistake that the policy around data visibility (including implementing encryption) is left at the control of the developers to manage via code; leaving developers manage security implementation is prone to long term errors and a high risk of inadvertent data breaches. Change management also becomes a lot more risky with a higher chance of mistakes happening when multiple entry points have to implement the same security. If your application requires the storage of data at any point (even if it is short-term storage) and that data contains any information that can be used to identify a person directly or indirectly, then managed data visibility is especially important; dismissing or not identifying the data controller role is a typical mistake made and is covered in further detail in Mistake 5.

HOW TO AVOID THIS MISTAKE?

The choice of databases is smaller for mobile platforms compared to traditional PC targets because of technical challenges around providing embedded database engines across multiple devices with a small footprint that can run within the capabilities of the devices and the restrictions imposed by some vendors around the use of external libraries. Due to limited choice, a great number of hardware vendors ship the open source database SQLite. SQLite doesn't however provide a way to implement a lot of good practice described. Encryption is not built in and it relies on external algorithms. It also promotes the security of data in the software layer rather than the data layer. You therefore have to look further afield to address in partnership with good practice the challenges of BYOD and CYOD around data storage.

Choosing an embeddable database like InterBase that supports multiple platforms, is transportable between those platforms, and supports the data layer containing the implemented data visibility policies reduces the risk associated with the data management around the application development lifecycle drastically. Additionally it frees up developers to focus on UI and performance requirements rather than becoming security specialists. This can also help with concerns around moving data to outsourced resources.

When implementing the data visibility policy inside the data layer a separate security login for granting and revoking access to data should be used to ensure the data security is separate from the data access security (like InterBase's SYSDSO login). This is very important and is ideally done in partnership with role-based authentication. By doing this once in the data layer, even if you code in different languages on multiple platforms, the data policy implementation for who can read what data is already implemented and ready to go. This simplifies aiming for a common codebase and reduced points for failure, that in turn reduces development costs and risks upfront and also as changes are requested and implemented.

Taking advantage of a full SQL92 compliant database that supports embedding business logic into the database via stored procedures helps drive efficiencies through test once – use everywhere capabilities. This reduces the testing requirements on the data layer to a large extent as the logic can be checked at source and then distributed allowing the process to have a quicker, lower cost time to market on all platforms.

From a security perspective, there must be a common encryption process across platforms. *Please refer to 'Mistake 5' for more detail.*

MISTAKE 5: RISKING REGULATORY ACTION AND LOSS OF CUSTOMERS BY TREATING SECURITY AND ENCRYPTION AS AN AFTER THOUGHT

Security is a major concern for application data in any organization, not just in the mobile world. However data security issues are exacerbated in the mobile environment. For instance, people are not always aware of the data protection laws relevant to their geographic area or the location they are shipping applications to (the law can vary from State to State in the USA for example). In short, any data that can be used to identify someone is a potential security breach and should be protected by 256bit AES strength encryption at rest either by recommendation or law.

Insecure data is a potential data breach, which can lead to regulatory action and fines. In reality though, the cost of a data breach goes beyond a financial penalty as a company typically loses three to four per cent of its customer base* following such an event. There are also additional associated costs such as credit checks, unforeseen discounts to customers to retain them and so on. There are repercussions for the executive team too – they hold the ultimate responsibility and so can be subjected to legal action.

HOW TO AVOID THIS MISTAKE?

Protection must start at inception

Security should be established across the product lifecycle – from the inception of the database, development and release, through to run on mobile, management and disposal. If encryption is not used from the beginning, then there is unsecured at-rest data! At any point in the application life cycle, there is a potential for data loss.

How often have we seen such events take place? – A developer gets given a copy of data to start a prototype application, he puts it on his USB stick, heads back to his desk, forgets to remove the data from his disk, gets into a taxi and loses the USB key. A similar story in the USA resulted in a $1.7m fine*. Similarly, how many times have phones/tablets been lost or stolen while travelling and the data is unable to be wiped remotely or a device is not reported lost until the employee returns to the office?

The only way that files can be made safe is through encryption and following a clear data access policy. It is a misconception that having a password on a device is adequate protection!*

The creation of PIA's (Personal Impact Assessments) as described by the ICO are a useful tool to help identify where protection may be needed and to help in identifying and managing risk. The ICO has a useful free guide available online covering this topic which is outside the scope of this paper.*

Empower the data controller; encryption and access rights

The role of data controller across the product life cycle is critical, but often overlooked or marginalized. The data controller's responsibilities include defining the confines in which an organization collects, stores, views and shares data. The data controller is also ultimately responsible for ensuring an organization adheres to its data usage policy.

Data storage must be encrypted with appropriate access rights configured before even a single record of data is added. To illustrate why, a company application may allow certain staff to access a list of employees and core details. Should one of these values be salary, then it would be important that only HR staff are able to see that data. The data controller therefore must enforce this condition in the data layer so only those with rights to view can see, search or edit the data in question – with access denied to even the development team! To this end, the data controller must set the rights to view data as defined by the company policy onto the data layer, which must be in the company's control and supported by user authentication.

Developers must not be responsible for implementing encryption.

Developers should not have to implement the wishes of the data controller. What if they forget or they have a bug in the code that exposes this data? Security must be in the data layer and any data collected should return default values where data is restricted. This way, even if the application is used inappropriately, the data is secure.

Avoid building the data policy into functional specifications

Data policy changes periodically – be it for legal, operational or other reasons. Therefore, data policy must not be built into the core logic of the application unless absolutely necessary and for a well-defined business reason. Again, data policy must be implemented by the data controller in the data layer. It is critical therefore that the restrictions placed on the data layer do not affect the functionality of the application. This is a role for testing. If functional specifications need to dictate the SQL statement to select the data, it must be regardless of the user connected, unless there is specific usability reason to do so. This helps manage risk in the development cycle and unexpected behavior towards application usage in the future. It also minimizes application support cost, application development is simplified and time to market is reduced.

In Conclusion

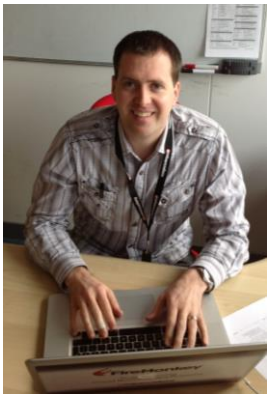
Enterprise application development is a reasonably mature field in IT terms and many companies are already following good practice. While mobilization brings a range of new challenges, the choice of the right database engine will support this practice and extend it into the natural evolution of enterprise data.

To manage costs around testing, to reduce risk around integration and data breaches, to ensure that data policy is adhered to, you need to close the net around what needs to be tested. This is easier said than done with BYOD and CYOD policies. The simplest way to manage this is to avoid the variations that exist with security settings and database options on different platforms with different configurations and use a specific embeddable database that allows this practice to be followed and one that works the same on all platforms.

By choosing a database/data layer that allows the data controller/data security team to be in charge of the data access, without any side effects for development, the data controller requirements can be contained as a side note to core functional specification document and provide the benefits for a secure deployment across the product lifecycle. New methods of keeping data caches updated are essential to explore to provide the best possible experience, scalability and cost saving.

InterBase covers the requirement identified above in terms of its encryption, multi-platform, and small-embedded footprint, and scalable change tracking capability. In addition to InterBase, the governance of metadata and the sharing of this data live time is easily managed though tools like ER/Studio Team Server from Embarcadero.

About the Author



Stephen has led development teams commercially for over 15 years within the UK and across Europe working with a range of blue chip companies including Hilton, American Express, Fitness First, Virgin Active; Stephen is a Charted IT Professional, a Senior Technical Sales Consultant (RAD) and the Associate Product Manager for InterBase, a product he has used commercially throughout his career. He is regularly speaking across EMEA and can be contacted via twitter or his blog.

twitter: @DelphiABall
blog: http://delphiaball.co.uk

REFERENCES & LINKS

Key definitions around data protection:
http://ico.org.uk/for_organisations/data_protection/the_guide/key_definitions

The cost of a USB lost that may have contained data
http://www.infosecurity-magazine.com/view/26630/

Cost of a data breach report
https://www4.symantec.com/mktginfo/whitepaper/053013_GL_NA_WP_Ponemon-2013-Cost-of-a-Data-Breach-Report_daiNA_cta72382.pdf

Guidance on PIA creation (ICO)
http://ico.org.uk/for_organisations/data_protection/topic_guides/privacy_impact_assessment

ICO Blog: Why encryption is important to data security
http://ico.org.uk/news/blog/2013/why-encryption-is-important-to-data-security

Change Views:
http://www.youtube.com/watch?v=Cnv7L36PqWE
http://delphiaball.co.uk/tag/change-views/

For more information on the Embarcadero products in this paper, please visit:
Team Server: http://www.embarcadero.com/products/erstudio-team-server
InterBase: http://www.embarcadero.com/products/interbase

ABOUT EMBARCADERO TECHNOLOGIES

Embarcadero Technologies, Inc. is the leading provider of software tools that empower application developers and data management professionals to design, build, and run applications and databases more efficiently in heterogeneous IT environments. Over 90 of the Fortune 100 and an active community of more than three million users worldwide rely on Embarcadero's award-winning products to optimize costs, streamline compliance, and accelerate development and innovation. Founded in 1993, Embarcadero is headquartered in San Francisco with offices located around the world. Embarcadero is online at www.embarcadero.com.