The Fusion Column - ODTUG Technical Journal - Q1 2011

Forms2Future or End Users on Strike

on Application Modernization (the Oracle way)

Early 2010, my company embarked on a marketing campaign, using the slogan 'Forms2Future'. Our plan was to appeal to organizations that had been using Oracle technology for developing applications for many years and that were looking ahead, trying to devise their strategy for current and coming custom applications. The slogan - Forms2Future - sort of backfired however. Since it mentioned Forms, people assumed it only applied to Forms applications, whereas we used Forms only as a label for all now dubbed *classical* development tools. More importantly, many people read it as 'from Forms to ...[someplace else]'. And assumed what we were talking about was a migration of existing Forms applications. Our original intention was nothing like application migration at all. A better preposition would be 'with' [Forms2Future]. And finally: '2Future' somehow suggested that the future is something that you can actually reach, with a single [big bang] journey. Whereas the future of course is very much a moving target that you keep targeting for, in a continuous, evolutionary process.

In this installment of the Fusion column, we will take a broad look at application modernization using Oracle (Fusion) technology. Taking into account industry trends, for example as captured by analysts such as Gartner, the rapid evolution of our end users - who in their spare time are browsing the internet using their mobile iSomething gadgets - and the changing business requirements in companies the world over, we will discuss what a modern application could look like. And we discuss how your applications - as well as your development staff and process - could morph into the modern world.

End users on strike

Recently I ran into a very explicit indication that modern times have arrived. I visited an organizations where the end users of an Oracle Forms application had gone on strike. They refused to use the application.

It turned out that these users had a simple, yet important task to perform for which they needed to use the Forms application. A dozen times per month, they had to decide on a proposal for a large purchase. To execute this task, they needed to launch the Client/Server Forms application on their local desktop. They had to navigate a vast global menu to locate the Form they required. The users then had to query the form using some primary identifier. For the decision they had to make, they require five pieces of data. However, the Form's canvas has been used to the last pixel - and even has a multi tab area. The five fields the user needs are all over the canvas and two different tabs. The decision is applied by changing the value of a dropdown list for the record's Status attribute. The users thought the application

was absurdly outdated, compared to their iPhone apps and the Web 2.0 applications they used at home and on the road for their personal affairs.

Figure - the experienced and hence demanding end user of today and tomorrow

And they were seriously disgruntled that such a simple task forced them to jump through all these hoops. That's why they were on strike. And had their secretary go into the application, print out the data, email it to them and finally record the decision based on their email reply or voicemail message.

All of a sudden, the approach that developers - including myself - had adopted for years, with powerful Forms that include fields for every column in the underlying tables and allow users to perform virtually any action on the data, seemed almost ridiculous, when looking through the eyes of these end users. They would be far better served by a user interface that presents just those five data items and two buttons - for Reject and Approve - instead of the information overload that they currently need to wade through.

That user interface should not require them to be seated behind their desktop in the office. Why not make it available in a browser, accessible on any computer including mobile devices. And instead of them having to find their way through a menu and the query the relevant record, the user interface could be pushed to them, from a Task List with To Do items or even by sending them an email with a deep-link hyperlink. Even more convenient: the email could be the user interface itself! The data needed to make the decision can be presented in the email and the decision can be applied by clicking on one of

two hyperlinks that either cause a reply email to be sent or directly invoke a URL on the application server.

Note however that in this same organization, there was a substantial user group working with the application on a daily basis for extensive data manipulation. These users were perfectly happy with the application as is. Even though they had noticed that it started to look a little outdated, it did not really bother them for the work they were performing.

This little anecdote is typical - I have seen or heard about similar situations in many organizations. We can learn several things from this example. Such as: Our user community is increasingly diverse - including consumers outside the enterprise, field personnel and commercial staff on the road, incidental business users and professional, trained, near-full time users. These users are more and more demanding, spoilt as they are by the Web 2.0 applications and social networking sites they are exposed to in their own time and the mobility they get from their *i*-devices. Lone gone are the days that Forms 3.0 applications on character based green on black terminals were considered modern - or even acceptable. User interfaces should closely support users in the task they need to perform -rather than just put a UI on a database table; this sounds logical - but can have serious consequences including the application's interaction design and the channel and device through which the user interface is delivered

However, today's and - more likely - yesterday's Forms applications usually still serve an important purpose for factions within the user community, typically those with the most intensive usage.

Aspects of Application Modernization

Triggered by recent developments in technology, evolving business requirements and demanding end users, updated Statements of Direction from Oracle or revised support and license policies, government regulations, compliancy rules, business partners looking for B2B interaction, uncertainty about mid-term job security, magic quadrant reports from analysts, inspiration from web wise multi-medial kids or other influences, IT managers and application developers in the most stalwart Oracle customer organizations feel the need for modernization of sorts.

Service Enablement

Whatever the shape the modernization will take, it is bound to involve multiple technologies, interact with third parties within or outside the enterprise and aspire to a substantial degree of agility. This calls for service enablement of enterprise resources - such as databases and content repositories - and adoption of some key SOA concepts regarding decoupling, interoperability, encapsulation and reuse.

Note that this does not necessarily mean the introduction of an Enterprise Service Bus and the widespread use of XML and Web Services, although this certainly may be helpful. Initially *service enablement* primarily means the use of well defined interfaces or APIs between clearly separated tiers. A key objective is reuse of functionality across technologies. This can be started with a View layer and PL/SQL APIs to insulate applications from the database tables and associated data oriented business

logic. The more data related logic - such as constraints and validation, security, auditing, calculation - is encapsulated inside the business services, the more this logic can be reused across all the user and programmatic interfaces that will be developed on top of it. In the near future expect each application (or cluster of functional interaction) to have a Service API in addition to a User Interface. That UI will of course most likely itself sit on top of that Service API. By the way: note that service enablement of both custom and standard applications will facilitate the integration and allow organizations to acquire Best of Breed applications for various segments of the business operations (a typical desire of business owners) and focus custom development on organization specific (competitive edge) aspects as well integration across the application landscape.

Products like the Oracle SOA Suite and Oracle Service Bus will make it easier to expose formal WebServices, both inside and especially external to the enterprise. They will also help you compose services that can be based on multiple resources, such as databases, content repositories and standard (commercial off the shelf) applications. In a landscape that has more than a single RDBMS for an enterprise resource, it will soon prove worthwhile to establish HTTP and XML based Web Services - leveraging Technology Adapters to interoperate with various technologies.

Workflow and Business Process Focus

Applications in the past often used to be departmental, associated with a single dedicated database and by and large a window on data navigated by end users through an extensive global menu. The applications did not necessarily recognize the purpose of the actions performed by its users, the role these actions play in the wider context of the business process.

Organizations increasingly recognize the fact that what they do for a living is not manipulate data, but execute business processes - however much data manipulation may be involved. Business processes can be described as strings of logically organized activities, ranging across IT systems, roles and departments. Some of these activities are automated and performed by machines, and some are 'humanual '. Then there are the configurable business rules that can execute complex business logic, taking over some of the human decision tasks.

An element in the modernization of applications is a focus on entire business processes that require partial implementation. The notion of an application itself is no longer very clear. The implementation of the business process consists of services that execute the automated actions and user interfaces that human staff use to carry out their tasks. But what are the boundaries of the application - and how meaningful is it to try to discern these boundaries?

Business Process Management is the architectural term coined for this focus on the process and BPMN (Business Process Modeling Notation) is the technique for laying down the design of the business processes. Oracle BPM Studio is a tool to analyze and model the business processes in a non-technical, business friendly manner. The combination of Oracle BPM 11g and SOA Suite 11g provides the technology to implement, execute and monitor both the process and its individual steps. All of this, based on industry standards and with support for "design time at run time" to allow for dynamic, business driven, on-the-fly reconfiguration and restructuring of business processes. In the past, Oracle

Workflow was the tool of choice for workflow driven applications. However, as of RDBMS 11g, Workflow is no longer supported as product in its own right.

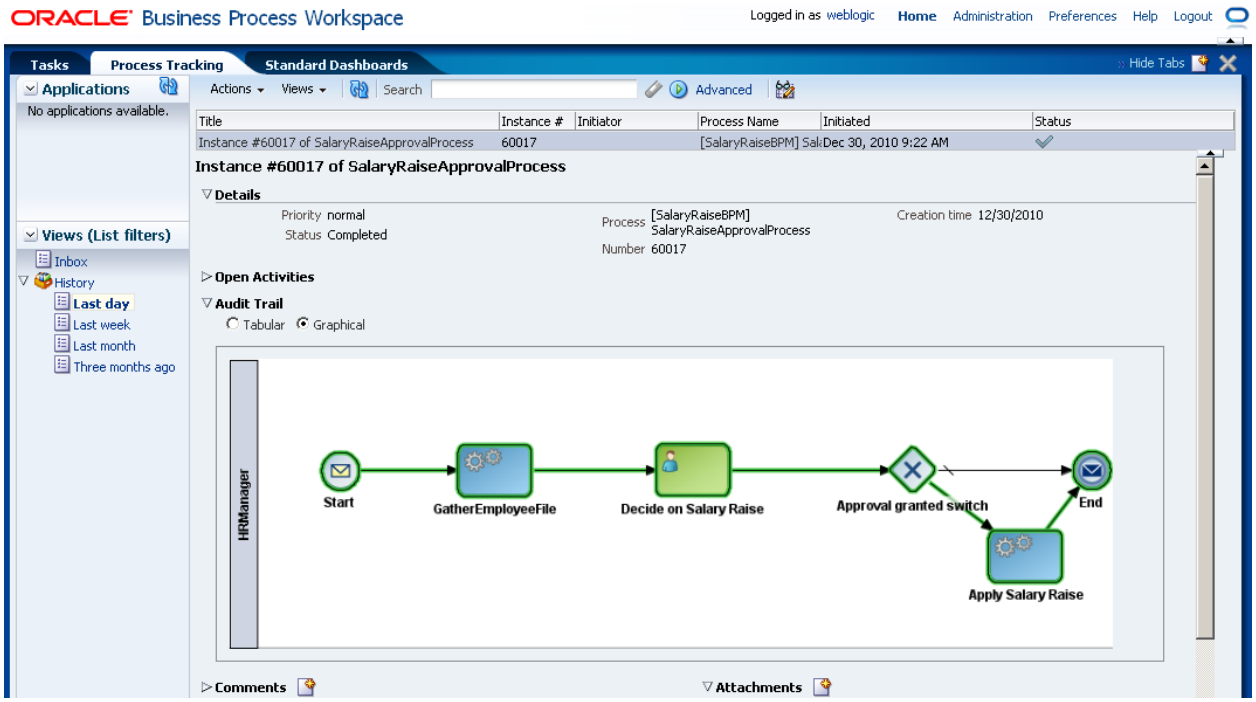

Figure - Screenshot from Business Workspace, the run time process monitoring tool for Oracle BPM

Even if you do not introduce BPM (right away), a focus on the business processes and central coordinator that organizes and monitors process instances is very useful. Like Oracle BPM, such an approach can provide real time insight in the process execution and offers real agility - the option to quickly change the process by rearranging and reconfiguring the individual steps.

Existing Forms , ADF or APEX front ends usually can continue to be used in a process driven approach. However, instead of being invoked from some global menu, these user interfaces may be triggered from a Task List and need to interact with the workflow engine to find out what data context they are supposed to present to the user.

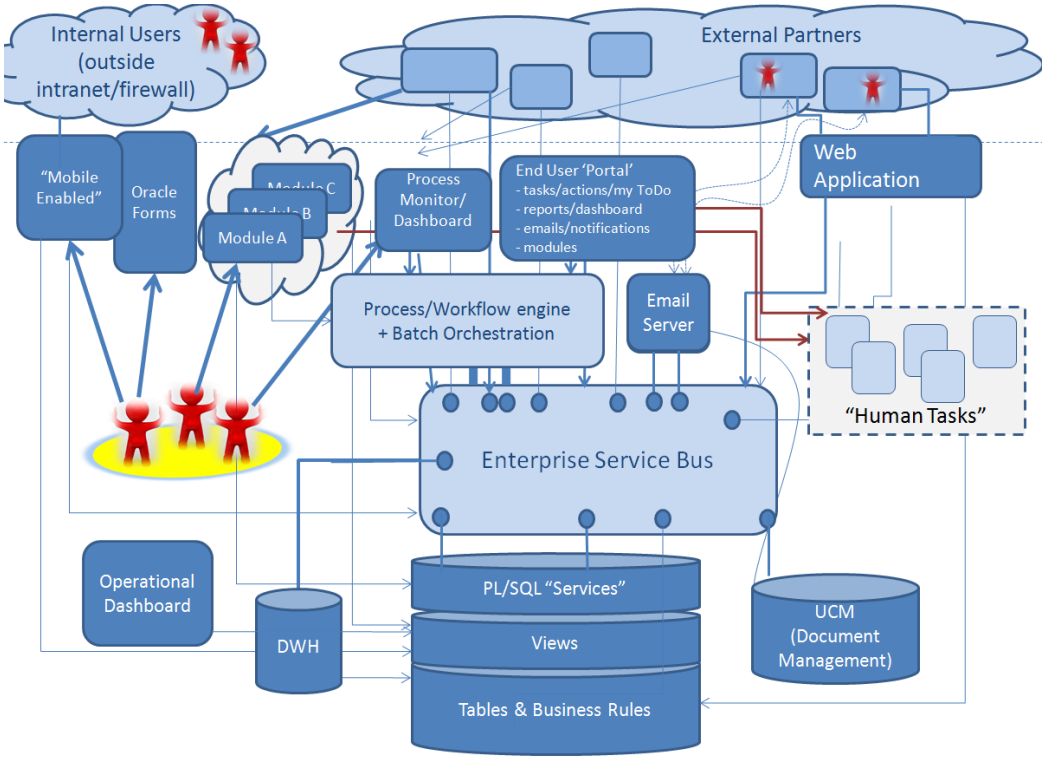

Figure - Overview of Modern Application Architecture

Agility

An important business objective is *agility*. To be able to quickly, effectively and with limited cost respond to changing requirements. Modern applications contribute to agility. Through a large degree of reuse, extensive run time configurability, highly decoupled systems that can be modified locally with low impact and small-scale, stand alone task oriented UI components - for example Portlets. Modern applications can turn IT departments into the enterprise's guide and inspirer rather than the limiting factor of old.

Note that agility is not just - far from it - driven by technology. Business agility much more depends on the interaction and mutual involvement between business and IT, the development process that should be focused on business priorities, short iterations and tangible deliverables and a software engineering process and infrastructure that caters for automated builds and tests, frequent releases and team collaboration.

Task Oriented User Interface - CRUD is dead!

A user interface in the modern world is not just a set of CRUD (Create, Retrieve, Update and Delete) operations on a database table - as has always more or less been the premise in Oracle Forms. Instead, the user interface is the interaction vehicle for a human actor to perform a task. Therefore, the UI should be focused on the task, provide all [and only] relevant details and guide the user in the execution

of the task, for example through wizards. This can be done using Forms, APEX, ADF and other UI technologies.

User interaction can mean much more than a screen with fields and buttons. Alternative means for interaction may be the most effective for particular tasks. Consider among others Excel (with ADF Desktop Intregration), email, voice response, SMS, Instant Messaging and touch sensors. Other important aspects that determine the effectiveness of the user interaction include the device through which the interaction occurs and the manner in which the task is presented to the end user.

Dashboard and Data Visualization

I have often wondered what it is that triggers a user to enter an application, navigate to a page, query a record and perform an action. I am sure it can be one of several things, like a phone call, some paper form, an email, a hunch perhaps. The process orientation and task focus make that most actions by users are instigated from a task list or some alert prompting the user into action.

Another angle that could drive a user's actions is the finding of an issue, an exception or serious deviation or pending deadline. Users with a responsibility for [part of] a process or a particular resource will need a dashboard that provides a real time overview of current status, action due, exceptions as well as recent and plotted trends. Data visualization that presents this overview and allows for *management by exception* is important - as pictures speak louder than words.

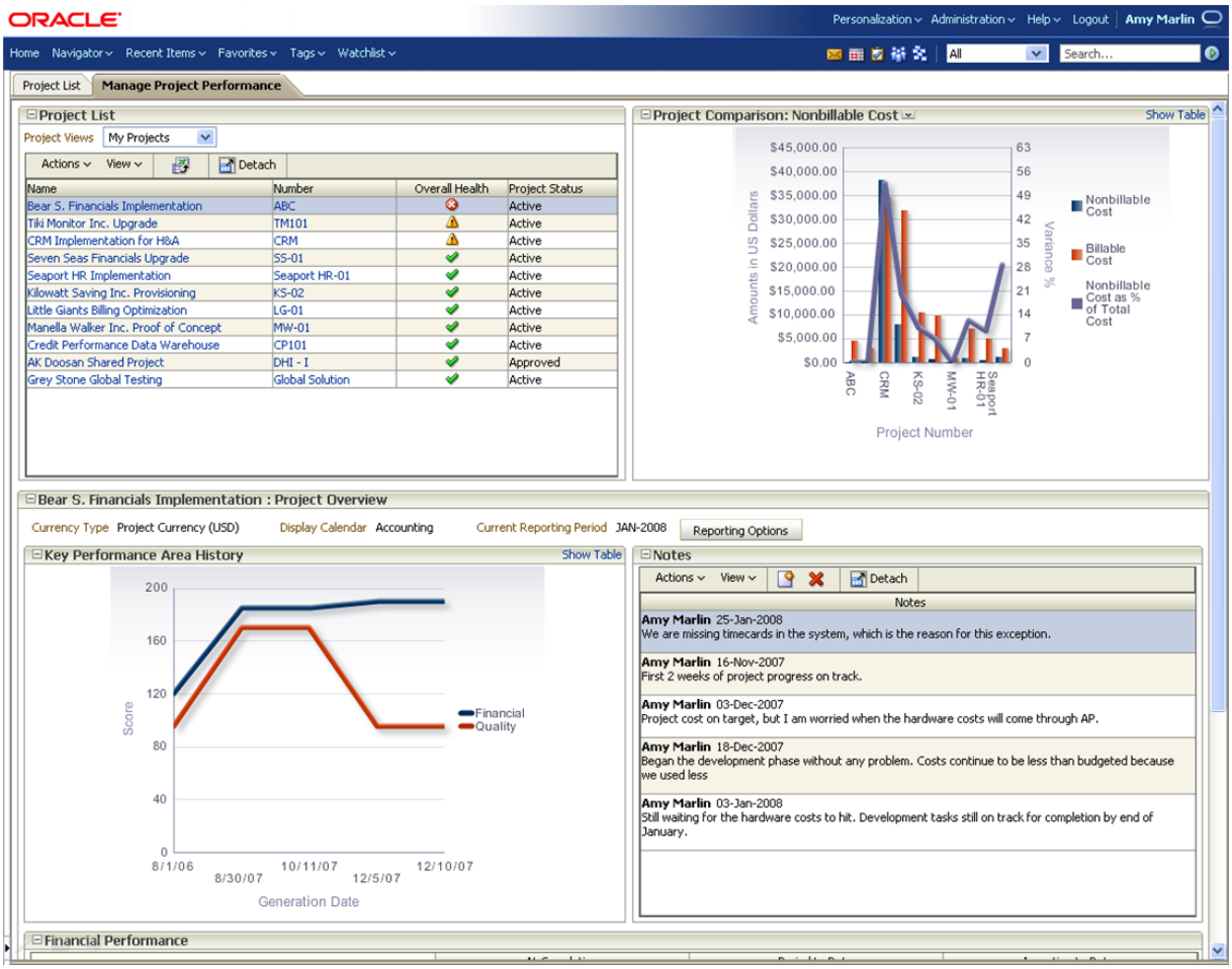

Figure - Screenshot from Fusion Applications, Project Management - visualizing trends, status, exceptions (in project) health and actions to take

A real time dashboard is also where the business process monitoring information can be gathered, such as BPM Workspace does for Oracle BPM. Exceptions clearly highlighted in the dashboard should provide the user with hooks to commence relevant (re)actions in response to the exceptional condition.

Oracle BAM (Business Activity Monitoring) is a tool to create a real time dashboard and send alerts when rules are violated. BAM's foundation is the active data model that captures and aggregates events and that is fed from sensors in SOA composite applications, JMS queues or other event publishers. ADF applications can leverage the BAM active data model through the BAM Data Control.

This BAM Data Control - along with controls for ESS Base, BI EE, URL, WebService, Pojo and ADF BC - in conjunction with the ADF DVT (Data Visualization Tags) can help create very rich, dynamic, interactive and animated dashboards. APEX too has data visualization support through the integrated AnyChart library.

In addition to the trends and exceptions that can be shown in dashboards, modern applications should also leverage existing knowledge to offer decision support and recommendations for actions. For

example through data mining of data on past behavior of all customers it is probably possible to predict the most likely courses of action for a specific customer and have the application advise on the appropriate pro active conduct. Another example: Oracle WebCenter ships with a recommendation engine that presents users with suggestions for content that they may be interested in.

When discussing dashboards and real time support, another element in modern applications should be mentioned: events. Publication, mediation and delivery of business and system events is one of the ways for gathering information used for monitoring and identifying exceptional situations. In environments with high volumes of real time events - for example regarding logistics, security, web traffic, sensor measurements, process execution - it often can prove beneficial to leverage Complex Event Processing (such as Oracle CEP) to sort out the thousands of events arriving every minute or even second and derive aggregates, find patterns and spot deviations. The event processor reports only meaningful findings, that can be pushed by the server to the dashboard.

Mobility - anyone, anytime, anyplace

Computing is ubiquitous. PDAs, tablets and even mobile phones provide more computing power in your hand than an entire server in the fairly recent past. End users are ubiquitous too - no longer are they office workers, tied to their desks. They can be field personnel, service engineers, account managers or consultants visiting with customers. Or doctors and nurses on the move through the hospital. They can be inspectors, truck drivers or waitresses. Anyone is a potential end user of our modern application, using that application at any given time from any given location. In modernizing our applications, it is important to cater for this. Task oriented user interfaces should be made available through the channel to the location and situation where it is required. Which may mean off-line aka in disconnected mode, for example in remote or secure or delicate areas where no wireless connection is available.

Another dimension that opens up with mobile devices is in the native functionality of the device itself: GPS, Photo and Video Camera, Microphone, Speaker, Email, Telephone, BlueTooth and InfraRed. End users and applications can leverage these functions in their interaction. The application for example can use the position of the device to push specific location based options or pre-filter selection lists. The user can add on site photographs to a report she is filing or scan barcodes.

Web Applications typically run on all mobile devices, in the local browser - in connected mode. The applications should cater for the smaller displays and device limitations with for example Flash and JavaScript (AJAX). APEX and ADF [Mobile Browser] applications can both be used from mobile devices. Note that Web Applications cannot tap into specific device features. Native applications - running as apps on the device itself - can work in off-line mode and can make use of the features of the device. These can be developed with ADF Mobile Client (for BlackBerry and Windows Mobile with Android hinted at for the future) in largely the same way as normal ADF Web Applications. The application architecture has to cater for application mobility, for example using light weight RESTful Http services and facilities like the Oracle Lite Mobile Server for synchronization.

The native email client on most mobile devices provides another user interaction channel with our modern application.

Power to the End User

The self conscious world [wide web] wise end user will not necessarily settle for the standard UI that is offered to every other user. He will expect user preferences to configure the behavior of the application and functionality to reorganize pages, redefine boilerplate text and add logos and other pictures. Standard applications, SaaS applications and even custom applications should offer these options for personalization to end users.

Run time adaptation of the application for entire user groups is usually called customization. Customization is called for when the application should be modified for all users in a certain role, department or country. This is typically done by an application administrator, in more or less the same way as end users do personalization.

Oracle has coined the term Design Time @ Run Time - to indicate the concept of modifying the appearance and behavior of applications after deployment, typically by content editors, application administrators and even end users rather than developers.

ADF has built-in features for run-time personalization and customization of pages and components, by users and application administrators. WebCenter - adds a wide range of features to dynamically configure, extend and redefine many aspects of ADF applications - through the Page Composer. The recent 11g PS3 release has broadened the list of elements that can be edited at run time to include Security, Page Template, Skin, Menu, Data Controls, Portlets and other page components, Page Creation and more.

Facilities for design time @ run time for the SOA Suite and BPM engine are available through the SOA Composer - for editing Domain Value Maps and Business Rules - and BPM Composer - to provide insight in BPM process definitions and to allow business analysts to document and edit these definitions on line.

Portal

In recent years the internet provided environments for social networking and community based collaboration that exceeded enterprise business applications in effectiveness, functionality and look & feel. It became a frustration for many users that the sophistication and ease of use they encountered while browsing the web and interacting through YouTube, Ebay, Twitter, Google Docs, MSN, Facebook and Twitter was sorely lacking from the business applications they had to use in their professional capacity. Not to mention gaming.

Applications used in the enterprise can benefit from mechanisms found to work so well on the internet. To accommodate the unhappy users certainly, but even more to leverage them in order to achieve more effective collaboration and communication and the establishment of communities within the enterprise.

Oracle introduced the term Enterprise 2.0 to describe the application of social networking to the Enterprise. WebCenter comes with a rich set of E2.0 services, such as tagging, linking, blogging, Wiki, Forum, Message Board, Instant Messaging, Presence and Email integration, People Connections (Linked In), ActivityStream (Twitter), Personal Profile (Facebook). These services can be used to construct a

portal framework - aka Mash Up - in which all task oriented user interfaces required by a particular user are embedded. The E2.0 services are available to the user wherever she may be in the application and whatever task she may performing. Access to colleagues [relevant to current task] is available all the time.

Despite OLE Containers and WebUtils in Forms, true integration of non-structured data has never really caught on in classic applications. However, it is clear that the majority of information used in business processes and human interaction is held in unstructured sources, such as images and video and PDF and Excel documents. User interfaces that support activities in business processes should therefore also cater for these unstructured resources. Oracle Universal Content Manager (UCM) is a product that provides a wide range of document services, including tagging, search, versioning, format conversion, archiving, scanning and imaging. UCM can be integrated into ADF applications through its Java API or WebServices or using the Document Services that are part of WebCenter.

An important element in this portal-framework is a powerful search capability, across all resources. Such a search should encompass structured data as well as many different formats of unstructured data. The meta data associated with the resources through tags, links, ratings, resource related activity and resource popularity should also be taken into account, as well of course as authorization rules. The ideal enterprise search facility is a combination of Google's pervasive search and Amazon's personal recommendations engine. As an example of such search capabilities, Oracle's Secure Enterprise Search can be integrated through WebCenter Services into ADF applications.

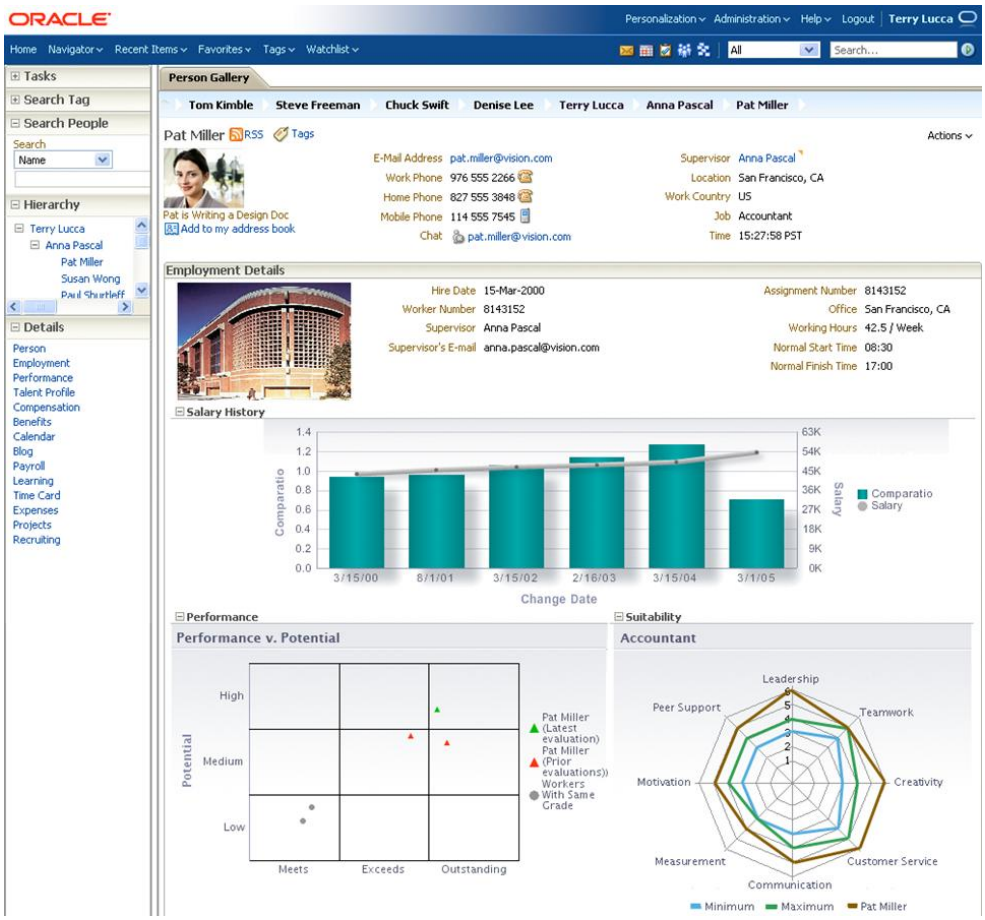

Figure - Screenshot from Fusion Applications - Web 2.0 and Social Networking applied to the enterprise

Cloud

The IT term of 2010 - apart from WikiLeaks perhaps - was probably *cloud*. So surely the cloud plays a part in the modernization of Oracle applications. And it does - possibly in several ways.

The cloud enables the smallest application vendors from the most remote corners of the world to sell their software as a service (SaaS). Web based applications running on public or private cloud infrastructure (IaaS)can be provided - to external or only internal customers. The model of a SaaS application is one that may very well be applied to any custom application, including those for internal usage.

Vice versa, organizations can easily enlist functionality from SaaS application offered by external providers, to complement or replace in house applications. One of the challenges the cloud will present developers with is the integration across the cloud - of custom and standard applications running on site with SaaS applications delivered from the cloud. Part of that challenge is the injection into SaaS applications of local services for areas like validation, calculation, emailing and printing - see the discussion in the previous installment of this column.

The cloud potentially offers a scalable, available and affordable infrastructure that can be used to facilitate distributed development projects - for example for source control, automated build management, stress testing etc. Note that APEX is in a league of its own as a PaaS solution it constitutes a Platform as a Service or a cloud based development platform.

Security

Huge challenges face us in the area of security. The number of individuals in our user community explodes when consumers and other outside parties become end user of our applications. Which means managing their accounts and protecting their identities and personal details. Additionally, we have to directly interface through B2B with systems of external parties such as suppliers, customers and government agencies - both outbound as well as inbound.

Our internal staff will work any-time and any-place, which is likely to be outside the firewall, outside the office space, from home and in the field. They access our systems through connections that should be secure and when data is stored locally on their devices, it needs to be appropriately encrypted.

And then our applications and data may no longer all reside in our secure environment - thanks to the Cloud and the SaaS applications and the IaaS infrastructure on which we run our systems. Privacy regulation, compliance legislation and anti-terrorist acts further complicate matters.

Conclusion

Many organizations are thinking about application modernization. And those who don't, really should. Not as a single big monumental step, but as a continuous process. To satisfy rapidly evolving business requirements leveraging the ongoing evolution of IT technology. This is called business agility.

Co-existence of user interfaces developed using various technologies with a specific task and user group in mind is the future. Architecture that supports such co-existence and interoperability is called for. Service orientation and a focus on decoupling user interfaces from the business tier is crucial in that architecture. Several components from Oracle Fusion Middleware can be put to good use in implementing the architecture that is the foundation for modern application.

Contrary to popular belief, application modernization is certainly not about getting rid of Oracle Forms based application as quickly as possible. Many Forms-applications still play a valuable role, especially for professional, trained computer users in enterprises. Migration of existing forms to other technologies is not an essential part of Application Modernization, and is even fairly pointless in most situations given the business objectives and requirements. Complementing existing forms with newly developed, task oriented user interfaces on appropriate devices and through fitting channels is much more sensible.