

The Future of Forms is... Forms!

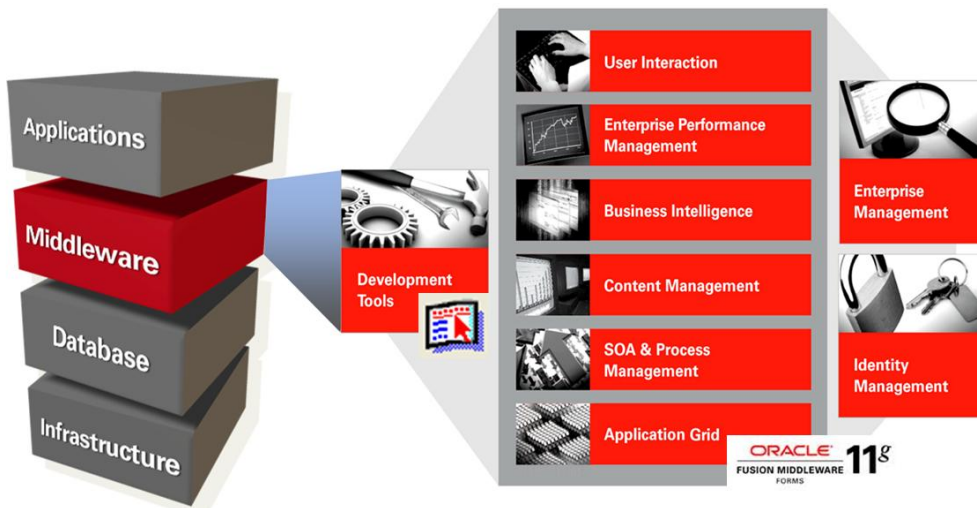
In a past installment of this column, I wrote about 'end users that went on strike' because they refused to make use of the Forms application in their company. I described in that column how a new generation of users - definitely outside the company but increasingly also within the enterprise perimeter - has new demands for the applications they use for their every day jobs. Used to social networks and popular web sites, working with modern devices such as iPhone and iPad and accustomed to media-rich, visually appealing apps and sites, they won't settle for the technology and look and feel from the '90s that is typical of most Forms applications.

This column prompted a riposte from Grant Ronald, Product Manager at Oracle for (among others) Forms - which in turn led to Grant and me presenting at the UKOUG 2011 conference on the future of forms under the same title of this column (see <http://www.slideshare.net/lucasjellema/the-future-of-forms-is-forms-and-some-friends-ukoug-2011-with-grant-ronald> for the slides from this presentation). Does this mean that I have misjudged the modern enterprise application user? That I have revised my opinion? Well, not as such. The title of this column is true, but it is not the whole truth.

The whole truth is that on the one hand Forms still has a lot of future left in it. It has fulfilled a particular job in the past and for quite some time to come it will continue to play that role, especially given the evolution it has undergone itself. However, certain new functional requirements, new user groups and interaction channels as well as new architectural directions will also bring in new technologies. Forms and these new technologies should integrate as good as possible, ideally share infrastructure and reuse application components. The better this integration, reuse and sharing, the more future there will be for Forms.

The Evolution of Forms

Forms is part of Fusion Middleware. This means for example that the Forms 11g releases run on WebLogic Server - which is clearly the strategic way forward in terms of application servers out of Oracle. Administration for Forms is integrated through Enterprise Manager FMW Control along with all other FMW components. The clustering, monitoring and management facilities available for WebLogic and FMW can be used for Forms too - and are a step up from the Forms 10g on OAS tooling.

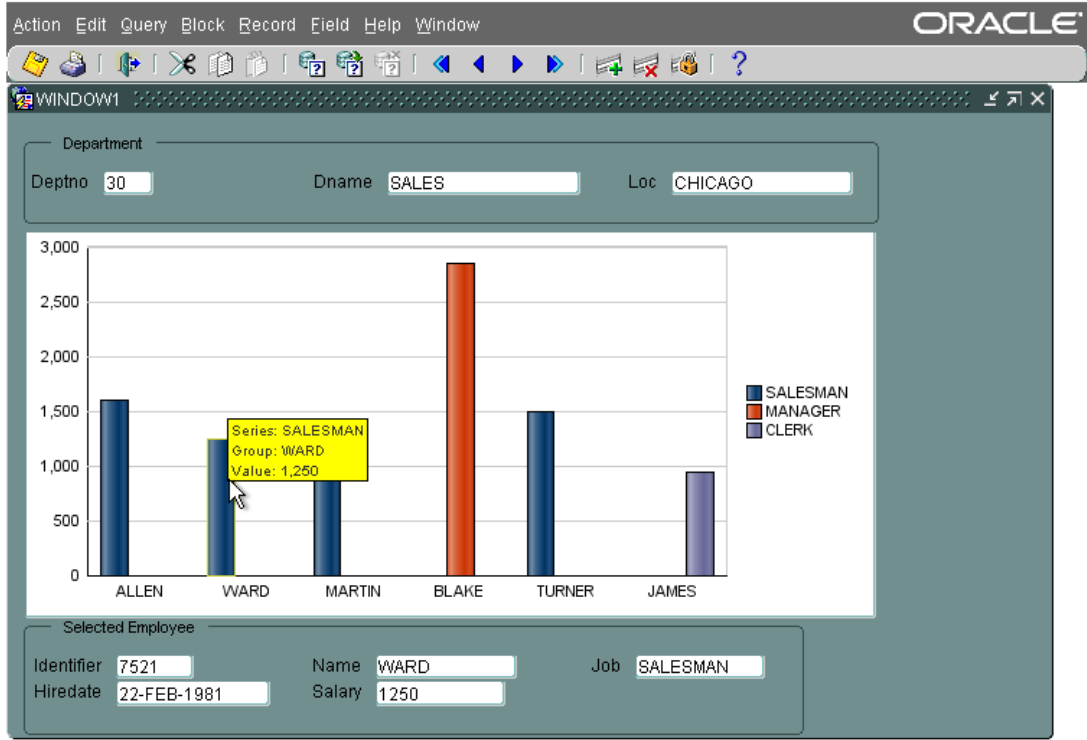


October 2011 saw the release of Forms 11gR2, and for late 2012 or early 2013 we can also expect the Forms 12c release. The evolution of Forms continues - albeit with fairly small steps. Support for Forms has been announced (extended support for 11gR2) until October 2017; we can assume that Forms 12c will have support until way beyond that date. The conclusion should be, that there are no technical or support related reasons for abandoning Forms.

The above statements demonstrate that Forms is alive. They do not make anything clear about the functional enhancements that have been made to Forms in the last decade. It is worthwhile spending some time on the most important functional capabilities. These help implement new functional requirements using Forms, modernize the look and feel of existing applications and achieve integration with standard technologies.

Java integration through Pluggable Java Components and Java Importer

Already introduced in Forms 9i and enhanced over the years ever since, Forms has strong capabilities regarding the integration of Java. The Java Importer on the middle tier allows the integration of Java Classes into the Form to leverage Java and JEE capabilities such as sending emails, publish to JMS or invoke a WebService. Even more useful is the Java integration on the client side, through Pluggable Java Components (PJC). The PJC can be used to create custom UI widgets, enhance standard widgets, support various user actions and render data in new ways. Make sure to look for inspiration on <http://forms.pjc.bean.over-blog.com/articles-blog.html> where you will find a large collection of PJC samples - most are ready to use - and a vibrant community of dedicated Forms followers. Many UI widgets you may know from rich UIs on the web or desktop are available or can be created for your Forms applications, through PJC. Examples include Google Map display, rich text editor, calculator attached to item, spell check facilities, PDF Viewer, auto completion behavior on text items, POP3 email reader, rich charts and graphs (Community Project FormsGraph)



One community project deserves a special mention: the Forms Look and Feel community project. This project shares a PJC library that allows tuning of the look and feel of your Forms applications, using CSS stylesheets - used for regular HTML based web application. The PJC's interpret the CSS instructions and apply them to the Form items. This allows you to manipulate colors, fonts, shapes and many other aspects of the Form and the items in them - enabling a UI with a look and feel that is quite far removed from the regular - and not all that modern and attractive - Forms look and feel. The next illustration gives a simple example of what can be done.

The image shows two Oracle Forms windows side-by-side, demonstrating different themes for the same application.

Left Window: Oracle BLAF Look and Feel

Oracle BLAF Look and Feel

Empno	Ename	Job	Mgr	Hiredate	Sal	Comm
7369	SMITH	CLERK	7902	17-DEC-1980	915	
7499	ALLEN	SALESMAN	7698	20-FEB-1981	1600	
7521	WARD	SALESMAN	7698	22-FEB-1981	1250	
7566	JONES	MANAGER	7839	02-APR-1981	2975	
7654	MARTIN	SALESMAN	7698	28-SEP-1981	1250	
7698	BLAKE	MANAGER	7839	01-MAY-1981	2850	
7782	CLARK	MANAGER	7839	09-JUN-1981	2450	
7788	SCOTT	ANALYST	7566	09-DEC-1982	3000	
7839	KING	PRESIDENT	17-DEC-1980	6000		
7844	TURNER	SALESMAN	7698	08-SEP-1981	1500	

Switch theme Ebay

Record: 1/14

Right Window: Ebay style Look and Feel

Ebay style Look and Feel

Empno	Ename	Job	Mgr	Hiredate	Sal	Comm	Dept
7369	SMITH	CLERK	7902	17-DEC-1980	915	10	20
7499	ALLEN	SALESMAN	7698	20-FEB-1981	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-1981	1250	500	30
7566	JONES	MANAGER	7839	02-APR-1981	2975	20	30
7654	MARTIN	SALESMAN	7698	28-SEP-1981	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-1981	2850	30	30
7782	CLARK	MANAGER	7839	09-JUN-1981	2450	10	30
7788	SCOTT	ANALYST	7566	09-DEC-1982	3000	20	30
7839	KING	PRESIDENT	17-DEC-1980	6000	10	30	30
7844	TURNER	SALESMAN	7698	08-SEP-1981	1500	0	30

Switch theme BLAF

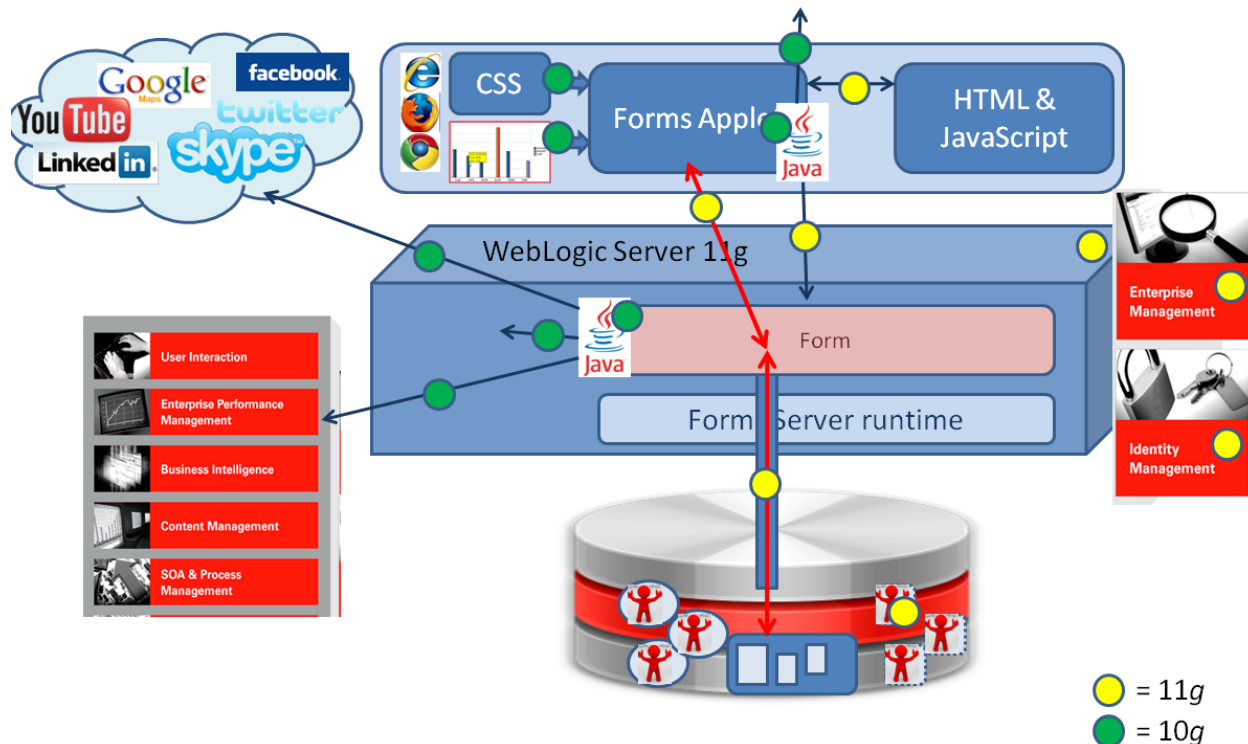
JavaScript API

An important enhancement in Forms 11g was the introduction of the JavaScript bridge. An API has been defined that allows JavaScript functions in the web-page that embeds the Forms applet to call into the form and at the same time enable Triggers and Program Units in the form to call out to JavaScript functions in the embedding web-page. This bridge makes it possible to create a client side mash up of Forms and HTML and JavaScript content rendered by other technologies, such as PHP, .NET, APEX or ADF. Events in one area of the page can be communicated to the form and vice versa which makes it possible to mutually synchronize and present an almost seamless user interface.

The JavaScript bridge opens up another interesting option: integration of Forms applications with the latest evolution in web technologies: WebSockets, one of the key constituents of HTML 5. WebSockets provides a lean and bi-directional communication protocol that allows for real push from server to client. Pushed messages are intercepted in JavaScript and can be forwarded into the form. In other words: Forms 11g applications can adopt WebSockets and thereby be push-enabled, fully participating in the latest development on the web. Note that even without embarking on WebSockets, the JavaScript bridge in combination with frameworks such as CometD allows a form of cross server push functionality.

AQ based push

In addition to the push discussed above which happens outside of the Forms Server, the 11gR1 release of Forms introduced push based on Advanced Queuing and handled by the Forms Server itself. A form can subscribe to events that are published as messages on an Advanced Queue in the underlying Oracle Database. This mechanism can be used to update forms and alert users as a result of events taking place in the database - or published to the database from other sources. It can also be leveraged to have on Forms-session (apparently) interacting with another. You could for example create a chat facility in Forms with users interacting via the Forms Server, AQ and the push facility.



Miscellaneous 11g improvements

Other recent improvements in Forms 11g include the ability to have a Forms application leverage the concept of Database Proxy Users. This allows users to be authenticated in the middle tier using accounts defined in Oracle Internet Directory. The proxy user - a single database account without any privileges used by Forms to connect to the database - can act on behalf of one or usually more database accounts with real application privileges. Using proxy users, You will increase the security of your application by separating the accounts which create database sessions from the accounts that can modify the database. In addition, the notion of every Forms end user requiring a database schema can be abandoned.

De recent 11gR2 release introduced integration with 11g Access Manager. Oracle Access Manager is an identity management solution that provides centralized authentication, policy based authorizations and auditing for all Fusion Middleware products. With 11g Release 2 you can use Oracle Access Manager for authentication and authorization of Forms applications.

Also new in 11gR2 are several features to allow more fine-grained recording of information such as which pages a user visits, how long they spend on those pages and even how long it takes them to perform a specific task, using Real User Experience Interaction (RUEI) - a feature of Oracle Fusion Middleware in Enterprise Manager that provides non-intrusive monitoring, giving insight into how a user is interacting with an application.

Just to be sure: JInitiator, the special browser plug in previously required to run the Forms applet has been abandoned long time ago. Today, a regular browser with the standard Java plug in - that most browsers not running on Apple's mobile devices already have installed - can run Forms applications.

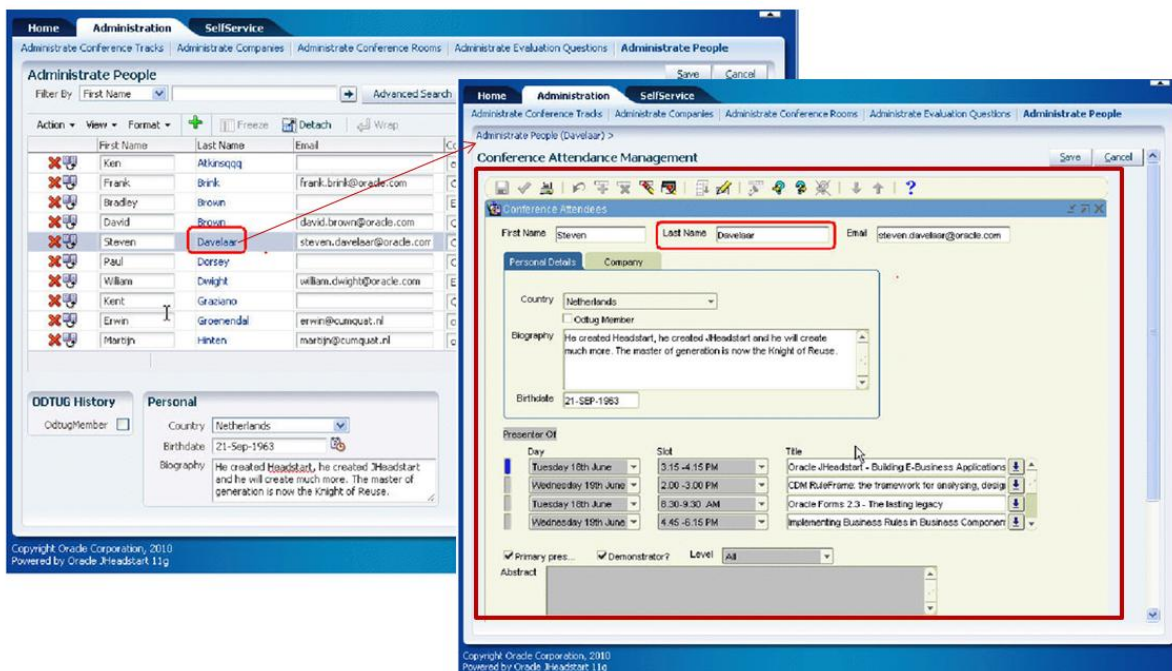
FoFs - Friends of Forms

Forms does not stand on its own. In the past, a Forms application with its own associated database may have been all you needed, but today's world requires integration of various technologies to together provide the required functionality. Additionally, organizations may well - and perhaps even should - choose to embark on a path to a more modern architecture and associated tool and skill set. Such a path can be evolutionary, adopting a hybrid environment for a prolonged period of time.

Through the bridges in Forms to standard technologies such as Java, JavaScript and AQ, integration and interaction with other components of Fusion Middleware and even 3rd party tools and technologies are a real possibility.

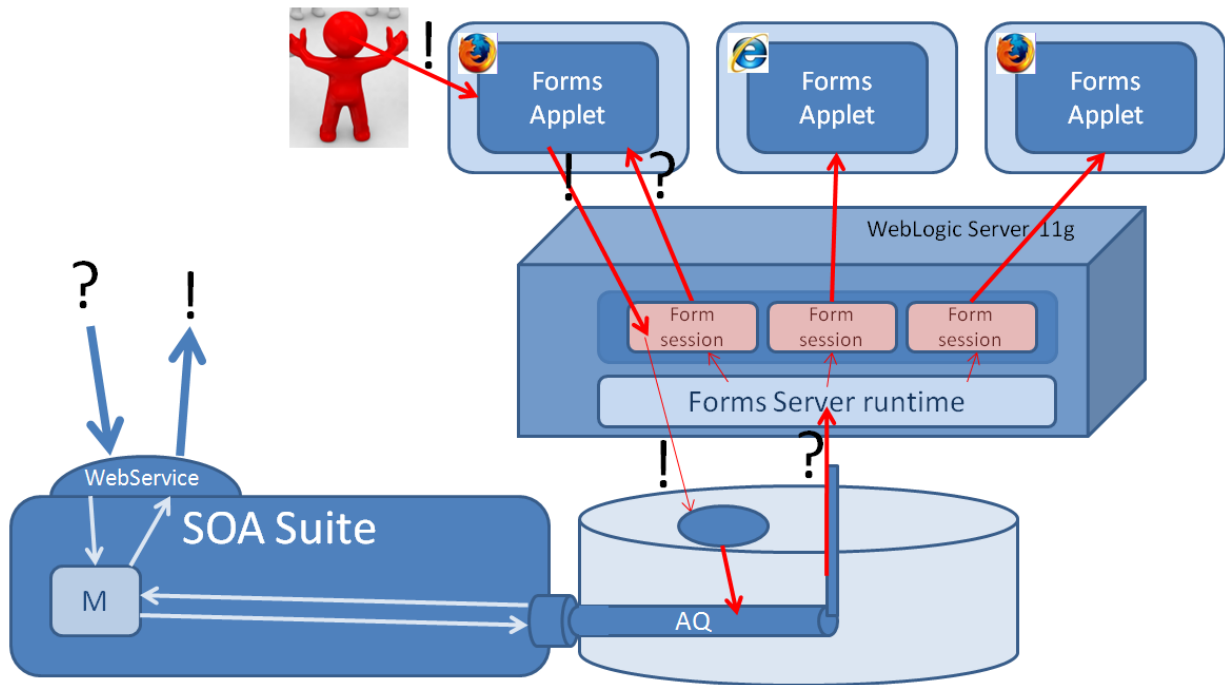
One example of this is the embedding of Forms in web applications created using other technologies including ADF, APEX and even PHP or any other web framework. Java Script is used to trigger the Form to set parameters, query the right data and otherwise synchronize with whatever is happening outside the Form in the part of the page that is governed by the web application. Vice versa, the Form will inform the web application of its doings, also through the JavaScript API. Navigation can be started in both the Form and the web application and apply to both parts.

This principle has been applied for a number of years in the OraFormsFaces product from Commit Consulting (<http://www.commit-consulting.com/oraformsfaces/>) proving the value of this approach for the combination of ADF and Forms. The next illustration shows an ADF application that embeds in one of its pages a Form. Navigation and the data context is triggered in the web application, the Form synchronizes accordingly.



Another increasingly common combination is Forms with SOA Suite - both part of Fusion Middleware, both running on WebLogic Server. Forms - through PJC's or imported Java Classes - can for example invoke the Web Services exposed on the SOA Suite. Additionally, Forms can use PJC's to communicate with the Human Workflow Service API in the SOA Suite and thus provide an alternative UI for human tasks that are instantiated in BPEL or BPM processes. The support for Advanced Queuing in Forms 11g introduce even more options for interacting, including some really farfetched ones:

The next illustration shows an asynchronous WebService exposed by the SOA Suite. A call to this WebService can be handled by a Mediator component that invokes the AQ Adapter to put a message on a queue. The Forms application can be registered on that queue. The message is pushed to one of the Forms sessions and can appear to a user in the form of a question that needs to be answered. The user keys in the reply, the Form posts that reply as a correlated message on another queue which is picked up by the AQ Adapter in SOA Suite that finally (and asynchronously) return the response to the caller.



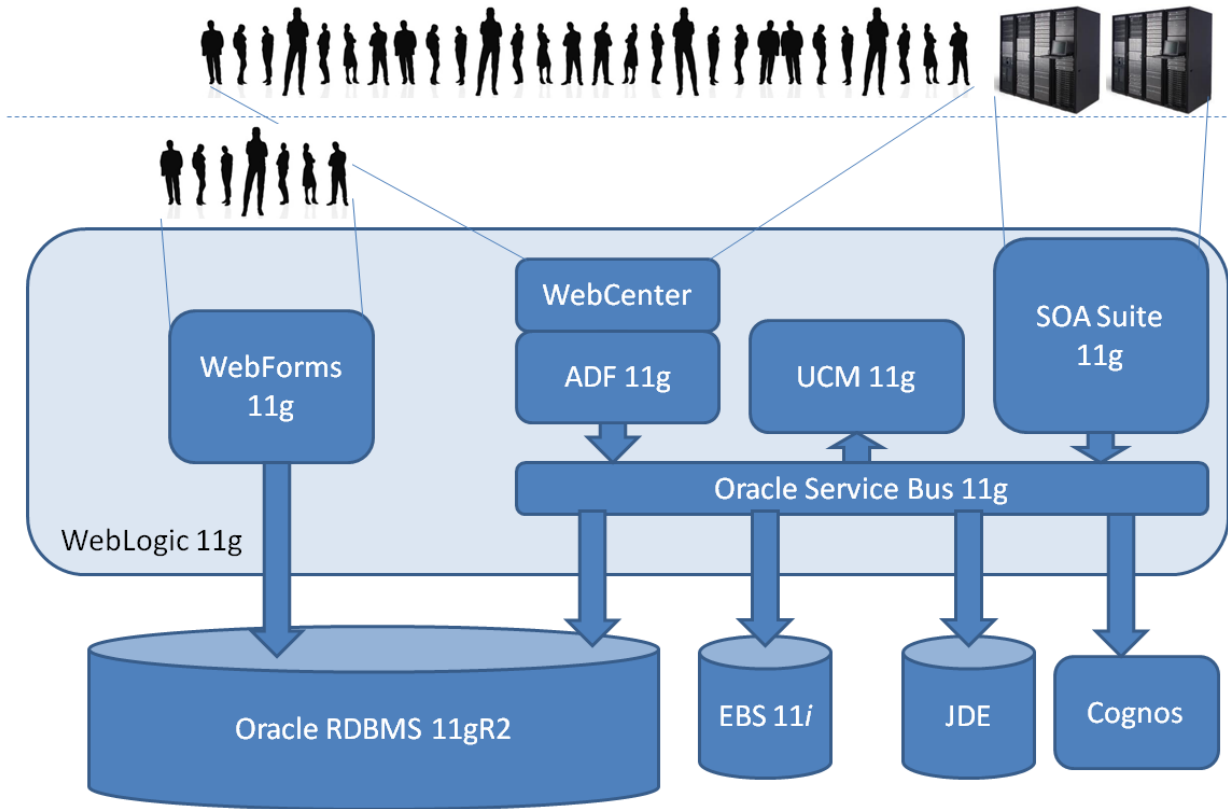
Note: this will not be an everyday scenario. However, it does show the capabilities of Forms to interact with 'the rest of the world'.

Moving Forms Forward

The future of Forms may be Forms, but that does not mean that nothing will change. Forms will not help you address new business requirements regarding mobile applications, external (self service) user groups, integration with standard applications - possibly cloud based - and cross application data integration. You should prepare for a future that consists of a hybrid architecture with a database (or more than one), your current Forms application (that can serve existing user groups for current functional requirements for some time to come - possibly with some refinements to the look and feel)

and new components such as SOA Suite, (extranet or internet) web applications and other enterprise resources such as content management systems, standard applications, BI servers and message infrastructures.

The next illustration shows an example of such a hybrid architecture as seen at one of my customers:



The Forms application, dating back to the late 90's is still very much in much for the professional workers, all on the internet and all specially trained on the application. They do the heavy lifting regarding the data and will typically use the application for intensive data manipulation for many hours per day. In the last few years, customers and business partners have gained access to data and operations, first through web services and later through an ADF based internet portal.

The team that worked on the Forms application since the end of last century is still doing that. They upgraded the Forms from Client/Server to WebForms 9i and later 10g, and now to 11g to benefit from the shared WebLogic based run time infrastructure and administration capabilities as the other FMW components they have deployed. An important task for the team has been to open up their database.

To call it *their* database is not really appropriate - even though that is how they feel about it. It used to be the database for the Forms application and it even has the same name as the application. However, the application is only one of many consumers of the database and the two - database and application - should really be considered two separate entities. This means for example that the assumption in the past that protecting data integrity - enforcing data oriented business rules - could be done in either the

form or in the database because it was all part of the same stack. With other components accessing the database - and not going through the Forms application - this clearly no longer holds true. Some of their business logic had to be transferred from forms PL/SQL program units to database stored procedures.

Additionally, blocks in Forms are typically based directly on tables or views at best. In the decoupled world that we strive for today, the services in the Oracle Service Bus that access the database should ideally not execute SQL but instead invoke a PL/SQL API that performs the SQL required. The team is currently working on PL/SQL packages that provides data services to the SOA team that implements the enterprise services in OSB. At this moment, the same team also works on the Forms application, but in the near future the team may well be split up into two , doing more justice to the clear split between the application and the database it leverages.

There may be even good reasons, in addition to allowing other to reuse the database under the Forms application, to have the application itself invoke services on the enterprise service bus and thus access other enterprise resources to add additional functionality to existing Forms applications.

Note that when we discuss Future of Forms and Forms applications, we should also discuss the Future of Forms developers. As you can tell from above, there is a future for Forms developers, because Forms as a development platform will be around for some time to come. However, I do not see many new applications developed in Forms or even major functional extensions to or rewrites of existing ones. There will gradually be less work in the area of Forms development in the near future.

Forms developers can evolve their own competitive edge - as well as providing added value to their end users - by investing in Java skills. In the short term, this will allow them to make use of Pluggable Java Components and Java Importer and in the long run, it will enable them to gradually move to other development technologies. Acquiring more HTML (5) and JavaScript skills is another valuable action for Forms developers: using the JavaScript bridge introduced in Forms 11g, a richer integration can be achieved of Forms in HTML pages. Additionally, these skill are valid for web and mobile applications created in any of a wide variety of alternative development technologies. Finally, it is my experience that many Forms developers - despite having been Oracle developer for many years - are typically not entirely up to speed with current capabilities of the Oracle Database, in both SQL and PL/SQL. It really pays off for developers to update their working knowledge of these core technologies.

Conclusion

The Future of Forms is.... Forms - or rather: a landscape in which ***Forms*** will probably play an important role for quite some time to come – along with new technologies to cater for new user groups, new channels and new functional requirements. Organizations may well continue to use their existing Forms applications - for current functional requirements and existing user groups. It is not easy to put together a business case for replacing these existing applications with applications offering exactly the same functionality - albeit in a different technology. Especially since if this replacement entails a divestment of the skill set of the development team. There is no pressing need - because of risk management, technical limitations or support reasons - to abandon Forms on short notice.

Major new developments should not lightly be undertaken in Forms though. More modern technologies are available that offer additional functionality, are more standards oriented and are more widely accepted and are more strategic to Oracle and as such have a far longer estimated life span and a more intensive evolution.