Statement of Direction

ORACLE FUSION MIDDLEWARE

Oracle JDeveloper & Oracle ADF Strategy and Roadmap

September 2006

Introduction

Oracle JDeveloper is a free Integrated Development Environment (IDE) covering the entire development lifecycle of Java EE and SOA applications. It offers out of the box an integrated tool set and design time to quickly build any style of application (web application, wireless, portal, database, business processes, etc.).

To further boost development productivity, Oracle Application Development Framework (Oracle ADF) is a standard Java EE framework aimed at simplifying the development of composite applications. Coupled with the visual and declarative tools that JDeveloper provides, Oracle ADF greatly simplifies application development and allows all types of developers to be more productive, while keeping the choices of technology or deployment platform open.

The combination of Oracle JDeveloper and Oracle ADF is not only used by thousands of customers but also is key to the development of today's Oracle E-Business Suite and of Oracle Fusion Applications, the next generation of Oracle's business applications. This statement of direction summarizes Oracle's vision and strategy for application development tools and provides a roadmap for the upcoming releases of Oracle JDeveloper and Oracle ADF.

Productivity with choice

The overall goal for JDeveloper and Oracle ADF is to offer the maximum level of productivity to all developers while keeping their choices open. In the past, increasing productivity meant vendor lock-in to a specific technology or platform. With JDeveloper and ADF, Oracle provides a visual and declarative development environment, boosting developers' productivity without tying them to a specific vendor but still offering them the choice of:

- **Technology**
  When building applications, a developer needs to choose between many different technologies for each layer of the architecture: for example, between JavaServer Faces (JSF), Swing or JSP for the view layer, or between EJB, Plain Old Java Objects (POJOs) or other productivity frameworks for the business logic and persistence layers.
  With JDeveloper and ADF it is not all or nothing. We appreciate that different requirements can be met with different technologies and we believe that it is up to the architect or the developer to pick and choose the technologies for the each architectural layers, according to their needs or previous investments. To accommodate this, JDeveloper and ADF support a wide range of technology combinations for implementing each application tier.
- **Development style**
  In most companies, the development team is comprised of developers from different backgrounds. For instance, some developers come from the 4GL world and are accustomed to a visual development approach, while other developers come from C or Java development with more experience in code-centric development.
  JDeveloper provides a single tool that allows different types of developers to work together, on the same application, by providing different levels of abstraction over the same application objects. Most development artifacts can be manipulated in a visual view, a dialog view or a code view. For instance, the JSF page flow can be edited either by dragging and dropping components onto it, by working with properties and dialogs, or by editing it directly in the XML configuration file. Whatever method you use, JDeveloper maintains the underlying file automatically, offering you the option of using the development style you are the most comfortable with.

- **Deployment platform**
  One of the main benefits of using open and standard technologies to build applications is the ability to choose the middleware platform you want to deploy the application to, providing portability between application server vendors. JDeveloper and ADF are built on top of open standards and applications built with this stack can be deployed to any Java EE application server and can access any SQL-92 database.
- **Development tools**
  Finally, even though we think that JDeveloper offers unparalleled development productivity, we recognize that developers might have already chosen other Java development tools. Again, given the standard nature of Oracle ADF, developers can choose to edit their Java code and maintain their applications in other IDEs, while still leveraging the runtime features of the ADF framework.

Complete and integrated

Another very important goal and design point for JDeveloper and ADF is to provide a development environment where all the tools and design time needed for all types of development and for all phases of the development life cycle are tightly integrated together, from modeling the requirements to deploying the application. Don't be misled by the "J" in JDeveloper: the tool is not only for Java development but has become the complete and integrated design time for the entire Fusion Middleware platform, allowing developers to stay in the same design time environment whether they are doing database, web, wireless, portal, business processes development, to name just a few capabilities, as well as to share the same concepts, terminology, and gestures, there by removing the need to install and learn new tools and thus greatly increasing their productivity.

A question that we are often asked is "Why hasn't Oracle implemented all of its design time features as Eclipse plug-ins?". Given the popularity of this open source IDE, it is a valid question. However, the answer is simple. By controlling the entire development environment, we have been the ability to produce both the shell and the core architecture which allows seamless integration of all types of design time environments, all sharing the same concepts and look and feel. If we just created specific plug-ins for Eclipse, each plug-in might be very effective on its own but it might not necessarily be consistent or compatible with third-party plug-ins, and would not offer to developers the level of productivity provided by a fully integrated environment. For those developers who have already made the choice to use Eclipse, we also contribute to this platform and lead open source projects to provide standalone Eclipse plug-ins for specific technologies such as JSF, EJB 3.0 or BPEL.

Similar to the Eclipse framework, JDeveloper architecture is very modular and every feature is actually an extension to the product This flexibility allows the developer to enable or disable specific functionalities, and offers an easy way for customers and partners to create their own extension to the product. A number of extensions are already available from the extension exchange on the Oracle Technology Network web site (OTN):
http://www.oracle.com/technology/products/jdev/htdocs/partners/addins/exchange/index.html.
To standardize the development of extensions among IDEs, Oracle is leading the JSR-198, "*A Standard Extension API for Integrated Development Environments*", which allows extension developers to build a single plug-in for all IDEs adhering to this standard. The latest releases of JDeveloper provide an implementation of this standard specification.

The foundation of Oracle Fusion Applications

JDeveloper and ADF are already used by thousand of companies worldwide, for building their next generation of enterprise applications, built on top of service-oriented architectures and Java EE technologies.

Oracle's strategy is to build the next versions of business applications with the same open standard technology stack that is offered to external customers. Therefore, JDeveloper and ADF is key to the development of Oracle Fusion Applications, the next generation of Oracle E-Business Suite, JD-Edwards, Peoplesoft, and Siebel applications.

All Fusion Applications are developed using JDeveloper and ADF, providing development productivity to more than 5,000 developers internally and offering out of the box the performance and scalability required for such critical applications.

Oracle's internal commitment to Oracle JDeveloper and ADF shows how critical this technology stack is strategically for the company. Moreover, it guarantees that this technology is proven for any type of development, including large enterprise applications such as Oracle Fusion Applications.

Versions and feature areas roadmap

JDeveloper and ADF 10.1.3

The current production version of JDeveloper (10.1.3) is focused on the following feature areas:

- Java developer productivity features such as refactoring, source control integration, significant code editor improvements, allowing JDeveloper to be best-of-breed in the Java IDE market
- Support for J2EE 1.4 and web services, as well as initial support for Java EE 5.0 such as EJB 3.0, Java Persistence API (JPA), and JavaServer Faces, allowing developers to take advantage of the latest standards
- Visual and declarative development, allowing drag-and-drop development for web, wireless and desktop applications, greatly improving developer productivity
- Ease of use and ease of development with ADF, introducing new technology choices (JSF, EJB 3.0), more productivity features, and new comprehensive documentation and developer's guides

For a full list of the 10.1.3 new features, please refer to the 10.1.3 new features document available from OTN: http://www.oracle.com/technology/products/jdev/collateral/1013newfeatures.htm.
Moving forward, our plan is to continue to add features to the 10.1.3 version by releasing 10.1.3 incremental releases. At the same time, we are working in parallel on the next major release, Release 11.

The upcoming 10.1.3.x releases, intend to focus on three main areas: productive and complete SOA development, portal capabilities in web applications and partial support for Java EE 5.0:

- Integrated support for SOA development
  The major features planned are: integration of the Business Process Modeler (BPEL) and design time, the Enterprise Service Bus visual design time and business rules editor
- Integrated support for Portal development
  Such as a new JSF portlet component allowing WSRP portlets to be displayed in any JSF page and the ability to deploy ADF Faces pages as WSRP portlets in a portal 10.1.4 environment or other WSRP compliant portals
- Initial support for Java EE 5.0
  Including support for the final EJB 3.0 specification and for the Java Persistence API (JPA) reference implementation

JDeveloper and ADF Release 11

Release 11 is the next major release of JDeveloper and ADF. The features planned for this release focus on improving developer's productivity for all types of development and all types of developers, and follow the strategy and design goals described above. The section below describes some of the feature areas for Release 11 and highlights some of the features planned for that release.

Support for the latest standards and open source projects
Oracle is strategically committed to lead, contribute to, and support open standards and open source projects. With release 11, we plan to continue to provide support for the latest standards such as Java EE 5.0, JavaServer Faces 1.2, EJB 3.0/JPA, JSR-227, and SDO, as well as new Web Services specifications.

On the open source front, our plan is to continue to support popular tools like Ant, JUnit, CVS, Subversion, and other design time utilities and to provide enhanced support for open source frameworks such as the latest version of Struts or Spring 2.0 to give some examples.

Development environment productivity

The goal is to provide a highly productive environment for a diverse set of developers and for all phases of the development lifecycle. Here are some planned features, which contribute to improved productivity within the development environment:

- Role-based configuration
  Allowing JDeveloper to be tailored to the needs of the user, based on a role specified at startup. Depending upon the user profile (database developer, business developer, web developer, etc.), only the relevant options, menus, and dialogs would be displayed.
- Resource Catalog
  A composite application can be built with, and can access, multiple types of resources (images, portlets, web services, documents, database objects, shared reusable components, etc.). The Resource Catalog intend to provide a federated view of the contents of one or more otherwise unrelated repositories in a unified search and browse UI, to enable users to quickly locate and consume application components from an arbitrary set of repositories.
- Dependency Analysis
  Allowing the developer to visualize how many objects in an entire application are dependent on a specific modification (such as removing a column in the database, for instance), reducing the risk of introducing new bugs when maintaining the application.
- Many design time additions and improvements
  Such as: improved user interface for navigators and editors, adding a new performance and CPU profiler, full support for JavaScript development, improved database features and schema modeling, and many other design time enhancements to improve the developer's experience.

Code sharing and reusability

An important aspect of application development is the ability to enforce a consistent user interface and to facilitate the creation of reusable code and composite objects across the team. In the Release 11, a number of features are planned to promote this type of development, such as:

- UI templates, to allow the definition of a standard user interface
- Declarative Components, to provide the ability to assemble user interface components together and create new reusable components
- Regions, to define a portion of a page as a reusable object and potentially expose it as a portlet
- Reusable task flows, to identify an entire page flow as one entity and to reuse this flow in different applications just by changing the input parameters

Interactive Web 2.0 user interfaces using Ajax and Rich Client JSF components

The strategy for the user interface is to provide the same programming model (JavaServer Faces) whether the developer wants to build a web application, a portal application, a wireless application, or a dynamic and interactive AJAX user interface. On top of providing a WYSIWYG editor and drag and drop facilities to easily build Web 2.0 interactive JSF applications, a number of features are planned in Release 11 to facilitate the development of user interfaces such as:

- An extensive set of AJAX/Rich Client JSF components intended to shield the developer from the complexity of JavaScript, while providing user interfaces that are both highly performant and interactive out of the box, by using open and standard technology (AJAX, JavaScript/DHTML). Using these AJAX-enabled widgets, developers should be able to build applications as interactive as desktop applications, running in a browser without any plug-in, while developing with a simple and declarative programming model
- JSF components for charting and graphing, with the ability to access both relational and cubic data sources
- Wireless JSF components with specific render kits for phone, PDA, or telnet devices
- Portlet JSF components allowing developers to render any WSRP portlet into a standard JSF page
- Active data channel for JSF components, which should allow the user interface to react asynchronously in real time to server events

Drag-and-drop data binding using JSR-227 data controls
JSR-227 data controls allow the developer to build a user interface that accesses data in the most efficient manner, and we plan to add support for more JSR-227 data controls to access new data sources, more user interface clients, and we are working on enhancements in the design time experience. Planned new features in this layer include:

- New types of data controls and data sources, such as support for not only relational data but also cubic data, web services, BPEL processes and other types of backend systems
- Drag-and-drop data binding for easily creating the databound user interface components in Swing panels, web pages, wireless devices, or portal pages
- A place holder data control to allow UI designers to prototype pages with dummy data that the developer can then wire to the appropriate data source when developing the actual application
- Creation of composite databound objects such as search regions or lists of values
- Integration with Microsoft Office to allow Office documents to also be databound using the same consistent data binding technology

Declarative business logic, data access, and security
There is a wide range of choices for developing the business services layer. In this release, we plan to support a number of approaches such as EJB, POJO, ADF Business Components. Our breadth of technology choices allows developers to select the most productive environment for developing the business logic and provides declarative features available independently of a chosen implementation. Some of the planned new features for business services development include:

- Declarative validation using standard expressions or scripting languages such as Groovy to simplify the development of business logic
- ADF Business Components improvements and optimizations (declarative way of exposing and consuming web services, brand new design time for improved usability, declarative definition of reusable lists of values, runtime performance and scalability optimizations, and so on)
- Productive development with the latest Java EE persistence standards (EJB 3.0/JPA), intended to provide declarative object-relational and object-XML mapping, annotation insight, test harness generation, and simplified deployment, to name only a few features
- End-to-end declarative security which would allow the developer to set standard JAAS authorization roles for the data and the user interface components, so as to more easily control at runtime what the user can access or display

Ease of deployment and management
To simplify the task of developers and administrators when deploying applications, Release 11 of JDeveloper plans to offer features allowing to:

- Further streamlined the deployment of standard Java EE archives and artifacts to any application server
- Package application code and metadata in one single archive, combining different type of artifacts together (such as portlets, business processes and web applications, for instance) into one deployable file
- Manage and monitor deployed applications using standard MBeans, offering the administrator a console to report resource consumption, connection pool information, and other type of metrics

The list of feature areas described in this document is of course not exhaustive and only highlights some of the planned capabilities of future versions of the product. A more detailed document will be available as new versions are released.

Conclusion

The current and future direction for the Oracle's development tools is to continue to provide more productivity to all type of developers, while supporting and promoting open and standard technologies. Oracle JDeveloper and Oracle ADF are key components of this strategy, providing customers with the tools and technology to build SOA applications faster while adhering to the best design patterns, and enabling the development of Oracle Fusion Applications, the next generation of Oracle's packaged applications.

Disclaimer

This document is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decision. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

ORACLE FUSION MIDDLEWARE

Oracle JDeveloper and ADF Statement of Direction
September 2006
Regis Louis

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com