ODTUG Technical Journal - Fusion Column - Q3 2011

Powering the cloud with Fusion Middleware

After careful consideration, I have decided to just do it. Write this quarter's installment of the Fusion Middleware column about a really trendy topic: the cloud. And more specifically: what Oracle Fusion Middleware can do in order to achieve the cloud objectives. However, before we get to that topic, it is imperative that I give our working definition of the cloud and its objectives in the context of this article. After all, clouds come in many different shapes and any author will morph the cloud to fit in with the story he has to tell. And to be absolutely clear about that: so will I. By the way: see this article for some amazing cloud shapes: http://scienceray.com/earth-sciences/meteorology/10-amazing-cloud-shapes/ or join the http://cloudappreciationsociety.org/.

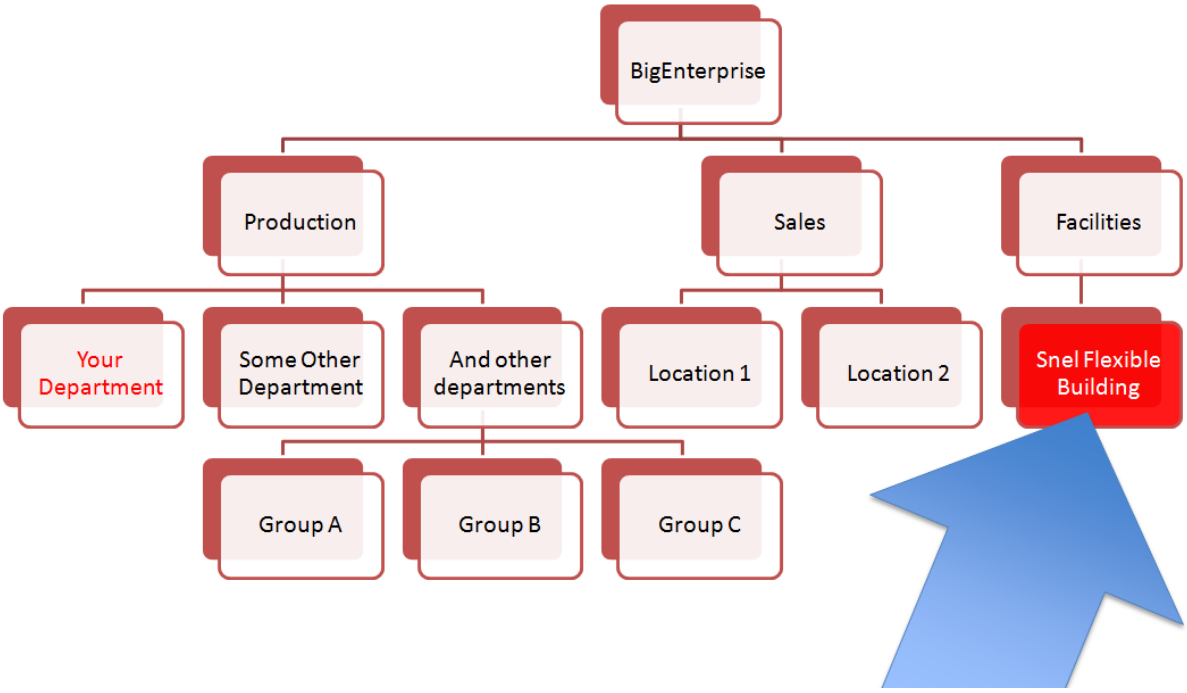
Flexible Building Services for Your Castle in the Cloud

Recently, in a local ad, I came across a company owned by a Dutch guy called Jan Snel. This company offers 'flexible building services'. If you need office space, they can build your 'castle in the cloud'. Using ready to deploy building units they can rapidly put your office together - fully furnished and plugged into facilities such as power, telephone, internet and water.

Some of the benefits of this flexible building approach

- No up-front investment
- Scale Up – quickly expand into new units
- Scale Down – have units removed when no longer required
- Offices can quickly be relocated to a different site
- Maintenance of the building is taken care of
- Day to day monitoring and security surveillance is provided
- Customers and standardize on office layout and interior across all branches
- Additional (backup) office units are available in case of emergencies (fire, flooding, …)

Large companies may decide to not rely on an external provider, such as the Snel Company, but instead create their own, embedded flexible office space department:



Clearly, for smaller companies such a private provider is not realistic.

Jan Snel happens to have two brothers: Piet and Simon. They too are in the business of providing flexible capacity. Piet provides the plot on which to build offices - including utilities electricity, water, sewerage as well as the drive way and parking spaces. Jan frequently works together with Piet for customers who

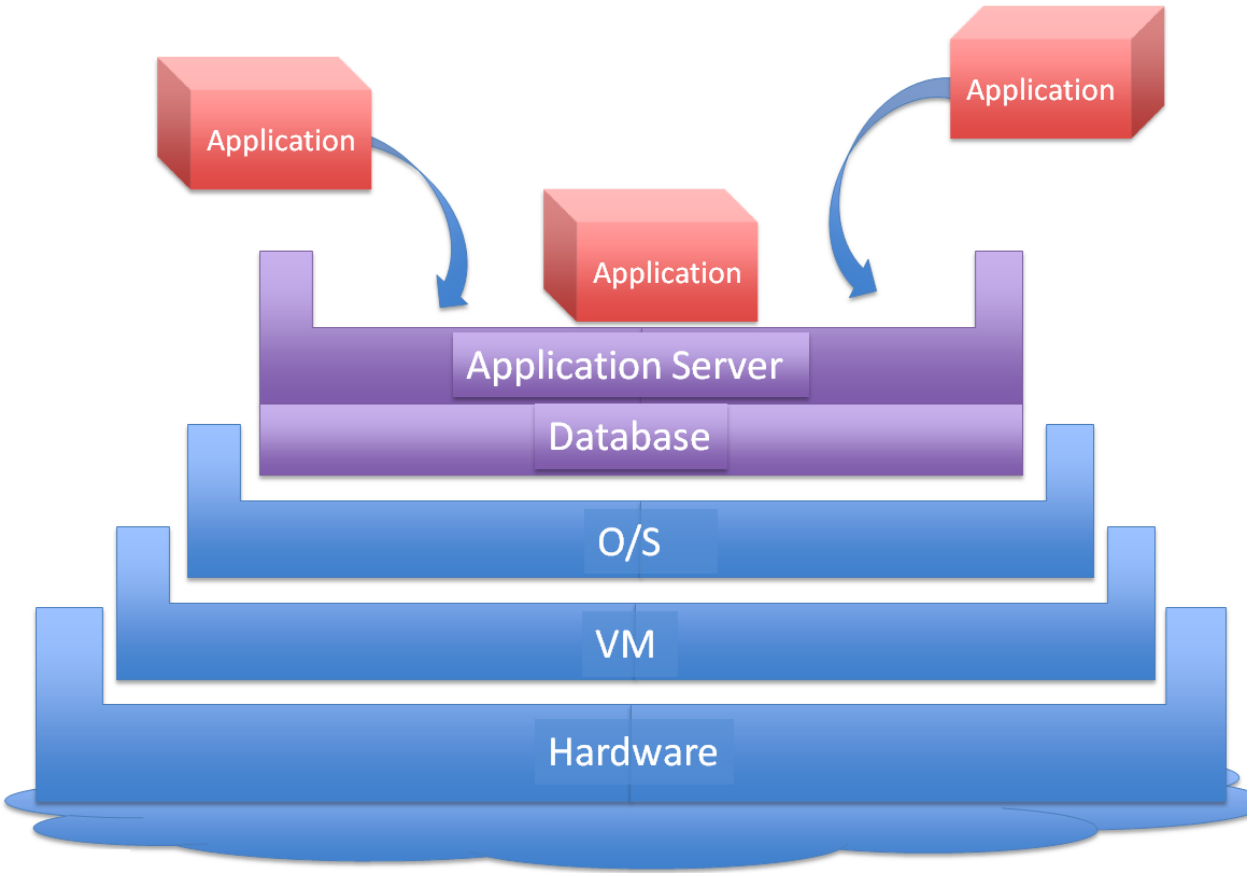
not have their own location to build the offices on. Some customers have their own building units and only need a plot to put them on; those will work with Piet only without engaging Jan.

The third brother Simon goes beyond what Piet and Jan can offer to customers. His approach is: if you need additional office space, then you will have people working those offices executing (parts of) business processes - by running a call center, staffing a sales desk, preparing shipments, handling complaints etc.. Simon says: "Why go through the trouble of organizing those additional offices, hiring and training people, doing day to day management of them and potentially having to fire them when the need for capacity decreases? Why not just engage my service center - that will simply execute those business processes. Perhaps they sit in my brother Jan's building, possibly on Piet's plot. Or in some completely unknown location. All you need to care about is their flexibly scalable output. With absolutely no upfront investment".

The benefits of flexible building listed above are readily understood in the context of temporary office space but can easily be translated to the world of cloud based IT services. Organizations can quickly get up and running with IT infrastructure, a deployment platform and ready to run software, without specialized knowledge, without upfront investment, with the ability to scale up (and down) and with professional round-the-clock administration.

The levels of functionality offered by the three Snel brothers also smoothly compares with the levels of cloud services: IaaS, PaaS and SaaS for Infrastructure, Platform and Software as a Service.

You may think these concepts a little abstract - IaaS, PaaS and SaaS. When we translate it to the actual components that constitute these levels, and look at the way these fit into each other as containers:



The infrastructure that a cloud can provide consists of hardware on which typically virtual machines will run that contain an operating system to run applications as well as storage devices for persisting data. Organizations may enlist infrastructure services from the cloud, such as Amazon EC2 or Microsoft Azure

or one of the many hosting parties that now offer their services in a cloud-oriented way- and subsequently install and manage their own platform on top of that infrastructure. Note that they can for example use their own licenses for Oracle Database en WebLogic Server on such public cloud infrastructures.

Organizations can also look for a PaaS provider - where the cloud partner offers a complete platform running on top of the infrastructure on which applications can be deployed. In that scenario, the organization will deploy standard applications or custom built applications on top of the centrally administrated platform that typically consists of a database and an application server. Examples of such public cloud based platforms are Google App Engine, CloudBees, Beanstalk and CloudFoundry.

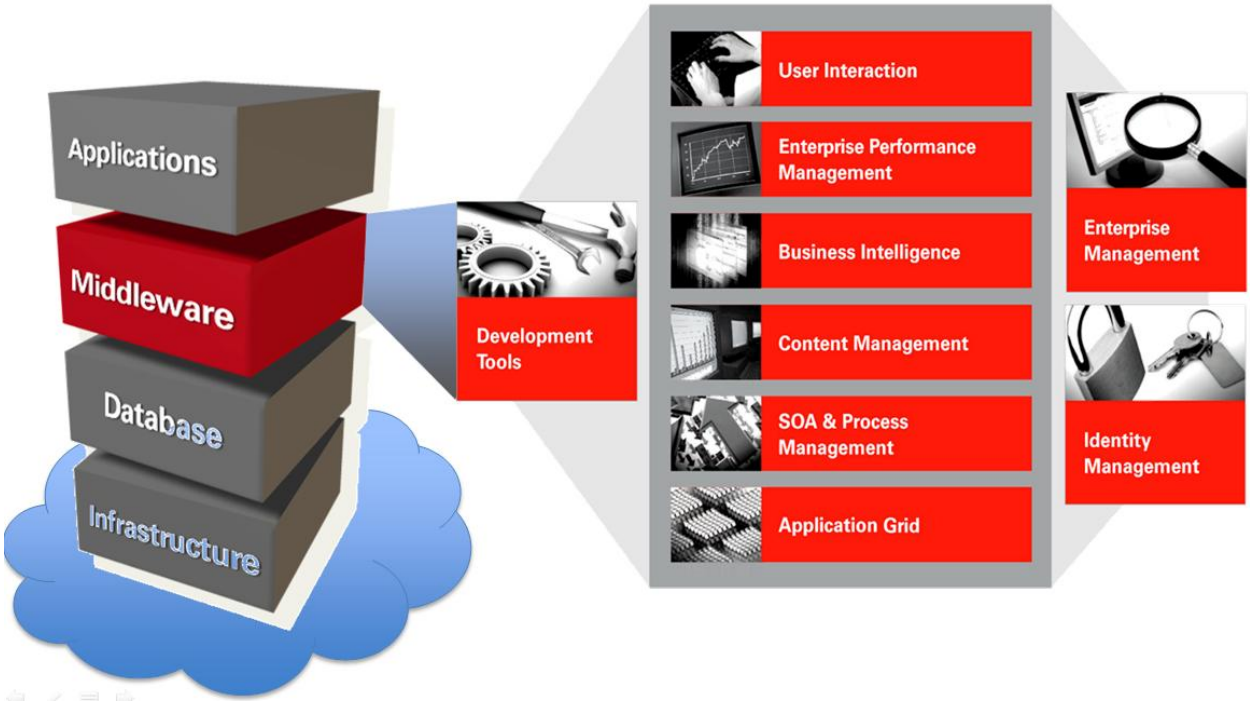
The next level of Cloud based services is SaaS or Software as a Service. In the case of SaaS - what a company engages is the functionality of standard software. The fact that this software runs on a platform that sits on some form of infrastructure is more or less irrelevant. The service level agreement an organizations has with the cloud provider dictates the availability of the functionality, the performance and scalability, the security of the data and the price per usage unit. Compare this with the offering made by Simon Snel: you do not need to have a plot (infrastructure) or even know where the site is of the office, you do not need the office and its facilities or even the people to put into the office (platform) - all you need to do is enlist the capacity to execute certain functions.

The level of Software as a Service typically contains the run time environment for applications. It can also provide a development environment in which applications can be developed - and later on built, tested and deployed to the run time environment. Development as a Service (or DaaS). The site apex.oracle.com can be seen as an example of Development as a Service. Another example of DaaS - for Java based applications - is CloudBees.

The Fusion Cloud

Oracle software can be used to power the cloud. When a company uses IaaS from a cloud provider, it can install Oracle Database and Oracle WebLogic to implement the platform. The infrastructure in the cloud itself can consist of Oracle (Sun) hardware - Exadata and Exalogic are prime suspects for this - and/or Oracle VM and Oracle Enterprise Linux. Oracle Fusion Applications are offered as a cloud based solution in addition to an on premise solution.

A cloud platform (PaaS) based on Oracle software is not necessarily a straightforward offering - in the public domain. Oracle is trying very hard to define a cloud oriented licensing strategy (including pay per use and flexible scalability) that brings in the margins its shareholders are used to. As a first tentative step, it is possible to engage the Oracle Database platform as a service with the Amazon RDS service (http://aws.amazon.com/rds/oracle/) with a pay-per-usage license option (currently only Standard Edition One). No such public PaaS offering is currently available for the Fusion Middleware platform.



Oracle's cloud strategy is focused primarily on the private cloud. With regards to flexible building, we discussed the option for very large enterprises to not engage a third party like Snel Inc. to gain the benefits of flexible building, but instead create an internal department that operates very much like Snel, but only for internal customers. Large organizations operate on a scale where for example internal resource demands from various departments fluctuate but the aggregate demand is more or less stable. In that case, departments can scale down or up without the enterprise scaling down or up. On the enterprise level, hardware and licenses can be more or less constant, while on the department level flexible scalability and pay per use is available. Such large organizations have enough volume in their central IT department to justify building up specialized skills, 24/7 administration and all the other faculties expected from cloud providers. Therefore, such IT departments can operate just like a cloud - a private cloud.

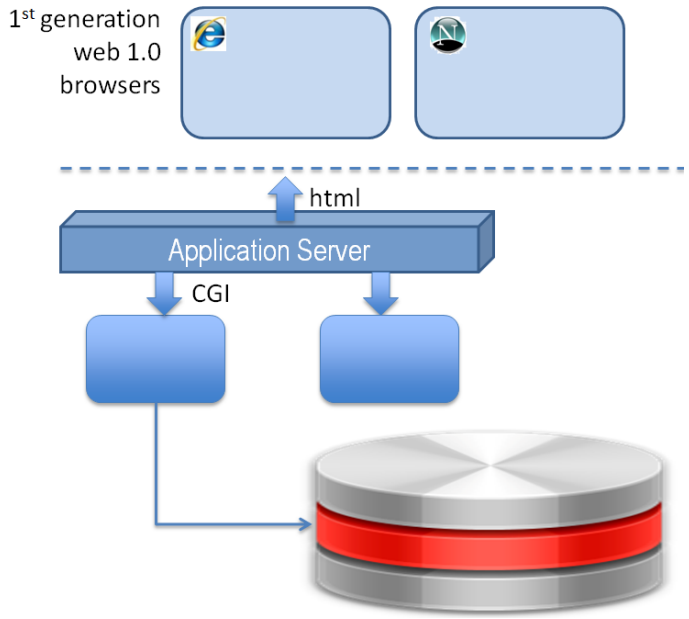
Note: Amazon's S3 and EC2 started out as internal IT departments for just the Amazon operations. Given the scale of those operations and the level of expertise and TCO attained by Amazon, it was realized that Amazon could offer IT services at much lower rates and higher quality levels than most internal IT departments at other organizations. Thus the private cloud evolved into a public cloud - before that term had come explicitly en vogue.

Fusion Middleware and the Private Cloud

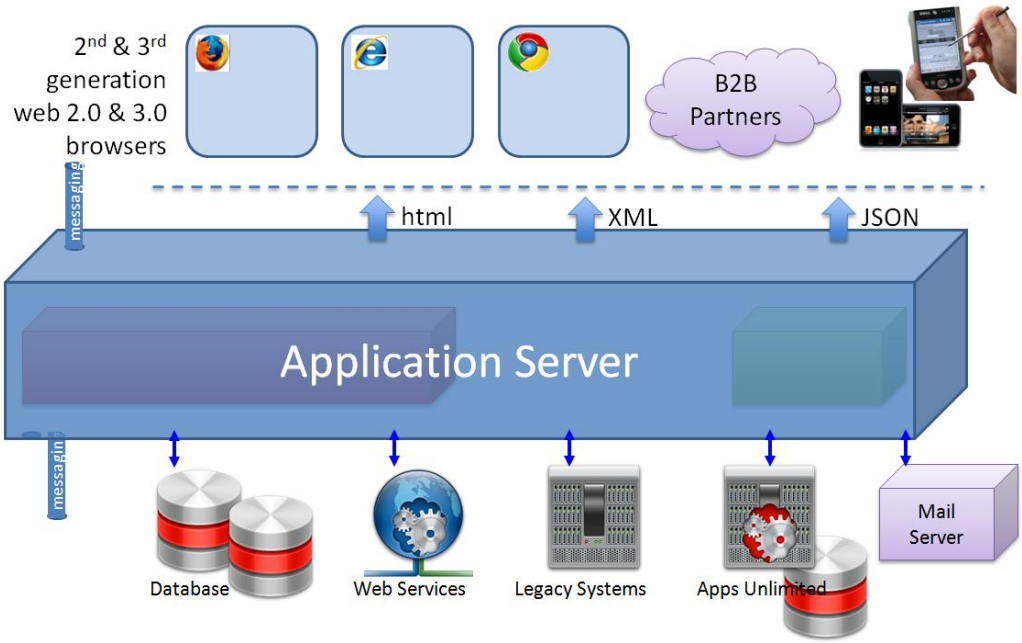
Well, no public Fusion Middleware Cloud Platform at the time of writing. Hopefully the shortly to be held Oracle Open World conference will shed some light on Oracle's strategy on cloud licensing. The claim that there are no technical issues preventing the use of FMW for a cloud implementation is proven by the many implementations of a Private Cloud platform powered by FMW. We will now take a closer

look at the question why Fusion Middleware is good for creating a cloud platform - private or public. To do so, we will first look at what we need from an application server that helps to implement a cloud platform and then at the features offered by WebLogic and FMW associates.

The role of the application server has dramatically expanded over the years. The term application server entered my world somewhere mid-90s. And initially this new kid on the block was a very flat character. As far as I could tell, the 'new thing in the middle', sitting between the end user application front end and the database did not much more than process HTTP requests, turning them into CGI calls to components that produce some result to return to or render as a user interface.

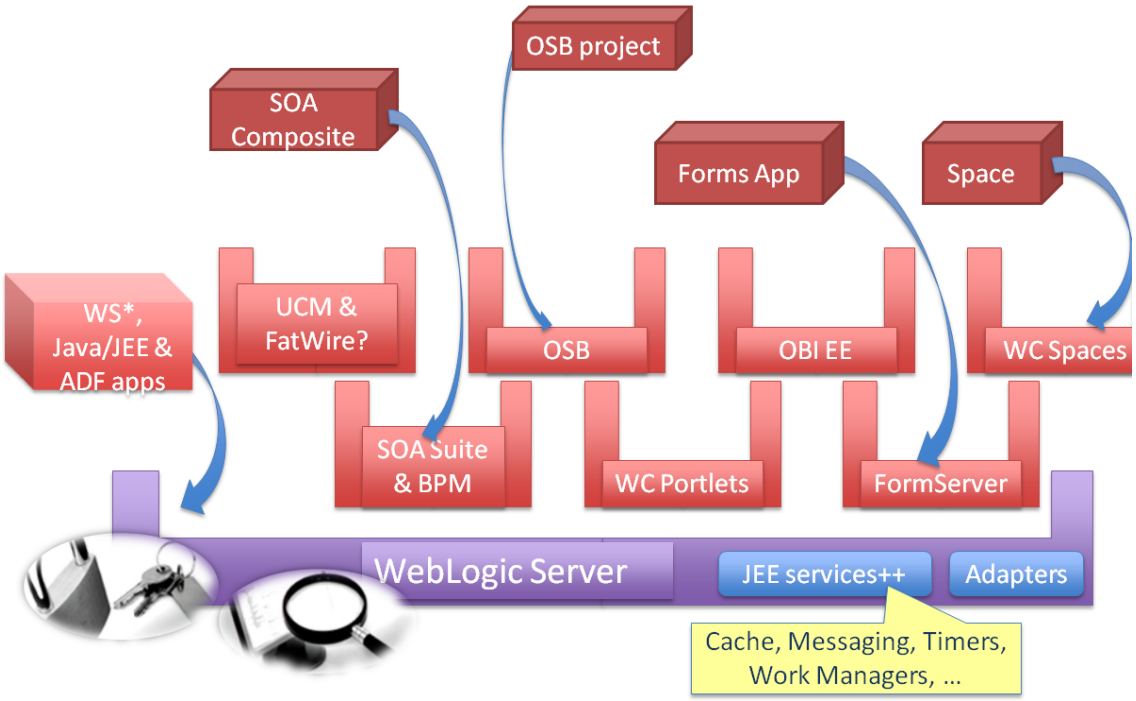


Gradually, over the years, the application server evolved - as a concept and as a product and technology component. The application server took on more responsibilities - for example around scalability and availability, security, caching, protocol adaption, service virtualization, messaging. The job of administering the application server no longer was something the DBA could take on in the hours between database administration - it became a highly specialized, demanding task that was crucial for the successful operation of the application landscape.



The JEE application server - WebLogic Server in the case of Oracle FMW - is inherently a container for running Java Enterprise applications. The container comes with the JVM (Java Virtual Machine) that executes the application and it provides a number of services to the application - dictated by the JEE specification - that allow applications to for example interact with databases, invoke authentication and authorization, leverage messaging and expose WebServices. However, the russian doll like hierarchy of containers does not stop with the application server as the ultimate container. Most Oracle Fusion Middleware products are deployed as JEE applications in the WLS container that act as containers themselves.

For example the Form Server is a container for running FMX applications that is itself installed into WebLogic Server. The Form Server benefits from the JEE services provided by the WebLogic container. Similarly, the SOA Suite is fundamentally a container for running SOA Composite Applications. This container is deployed as a JEE application on top of WebLogic Server and leverages the services provided by WLS, such as the adapter framework, JDBC Data Sources, EJB, JMX and platform security services. When you look from up close, the SOA Suite run time engine really contains a number of containers itself: the engines that run service components of types such as BPEL, Mediator and Human Task.



We have now established that an application running on the FMW platform is actually running as the pinnacle of a many-tiered pyramid of containers - that stretches from the underlying hardware (that is if we ignore the actual physical facility in which the hardware lives and the electricity that it consumes) all the way to the engine that runs the FMW component. Now the question should be: what is required of this stack of containers in order to make it realize the promise of the cloud. Or at least to allow a central IT department to deliver the cloud promise to its consumers.

Among all the benefits we are led to believe the cloud will deliver, there are a few that really stand out. The cloud should give us the ability to rapidly engage IT capacity, to quickly scale the consumption of that capacity, to pay per actual usage of that capacity and to have the capacity delivered in standardized format at predefined service levels with a low(er) TCO (total cost of ownership).

In the case of PaaS that we are discussing here - the IT capacity at stake is the ability to run applications at varying volumes of sessions and transactions with the maximum response times and the overall availability as defined, while adhering to compliance regulations. The WebLogic Server is central in the ability to deliver on this requirement of the Platform as a Service. We need it to be scalable, performing well, available, manageable in an efficient manner, secure and compliant. And of course it needs to provide the functionality to run the applications in the first place.

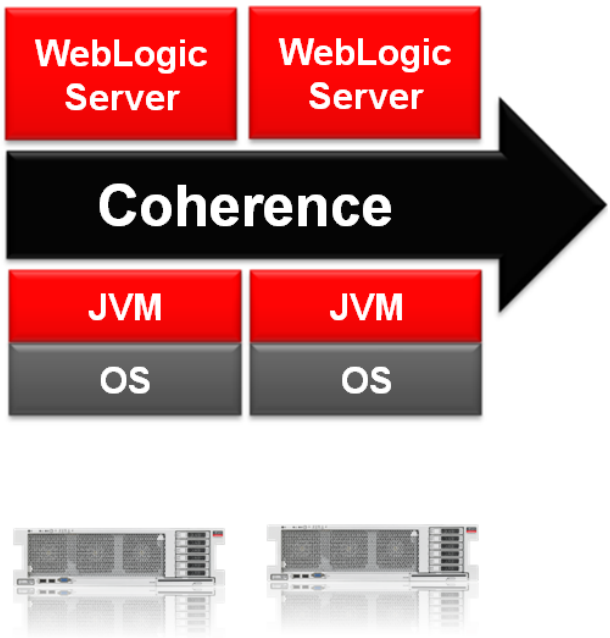
The WebLogic platform has proven itself extensively when it comes to its ability to run JEE applications as well as Fusion Middleware application components such as ADF, WebCenter Spaces, SOA Suite and OSB services, while providing platform security services and all JEE 5 and most JEE 6 services. I will not further discuss the functional capabilities in this article, but instead focus on specific aspects of WebLogic that make it suitable as a Cloud Platform.

Administration: most of the administrative tasks - including deployment, configuration and monitoring - can be performed through a browser interface. The web based Enterprise Manager Fusion Middleware Control console in combination with the WebLogic Server console make life fairly easy or at least straightforward for platform administrators. Additionally, many tasks can be automated using scripting (WLST). Using JMX and MBeans it is possible to configure applications and components by and large at run time - without the necessity for a redeployment or a restart. Compared to earlier generations of application servers and middleware components, the current components are very well integrated in the FMW stack. The deployment, configuration and subsequent administration is consistent throughout the platform. This promotes a low TCO and high availability. Flexibility and agility is enhanced through the fact that bringing new applications to the platform is a well defined process that can easily be executed - without impact on already deployed applications.

Scalability: the crucial cloud objective of providing flexible scalability as well as performance and availability is served well with the clustering capabilities in WebLogic. A WebLogic domain can contain multiple clusters that may consist of two or more managed servers (the lowest level container in which applications and components are deployed), each running on a different physical node. WebLogic performs load balancing of requests over all nodes in a cluster. Node failure is detected and requests are rerouted to available nodes. Furthermore, deployment of application is performed to a cluster and automatically rolled out by WebLogic over all cluster nodes. Adding nodes to or removing them from a WLS cluster is an easy operation that does not disrupt operations nor impact availability, hence provides flexible if not stepless scalability. More nodes means more fail-over options resulting in higher availability. More nodes also means higher performance through a greater (parallel) capacity for processing requests. Note that adding a node to a cluster may have a direct impact on the required licenses - an impact that does not necessarily scale down as flexibly as it scales up.

WebLogic through its GridLink facility can also work seamlessly with Oracle Database RAC: GridLink enables WebLogic to load balance database requests over the nodes of the RAC cluster. It can also detect the absence of a specific RAC node and automatically and transparently fail over to another database node.

Even though a cluster increases capacity and availability, this increase is sometimes not as great as one would want. When the cluster nodes do not perform constant synchronization, loss of a node will typically have a noticeable impact on users and their sessions. And when the nodes do fully synchronize, this synchronization may take up a substantial percentage of the processing capacity of the cluster nodes. The memory grid component Coherence can be used in WebLogic to share session state between cluster nodes without the overhead of inter-node synchronization. Use of Coherence for saving session state provides considerable benefits for a non-clustered WebLogic server, because of the more efficient memory usage especially for large session objects. In order to use Coherence*Web and ActiveCache, only WebLogic-level configuration is required. The application does not require any modification.

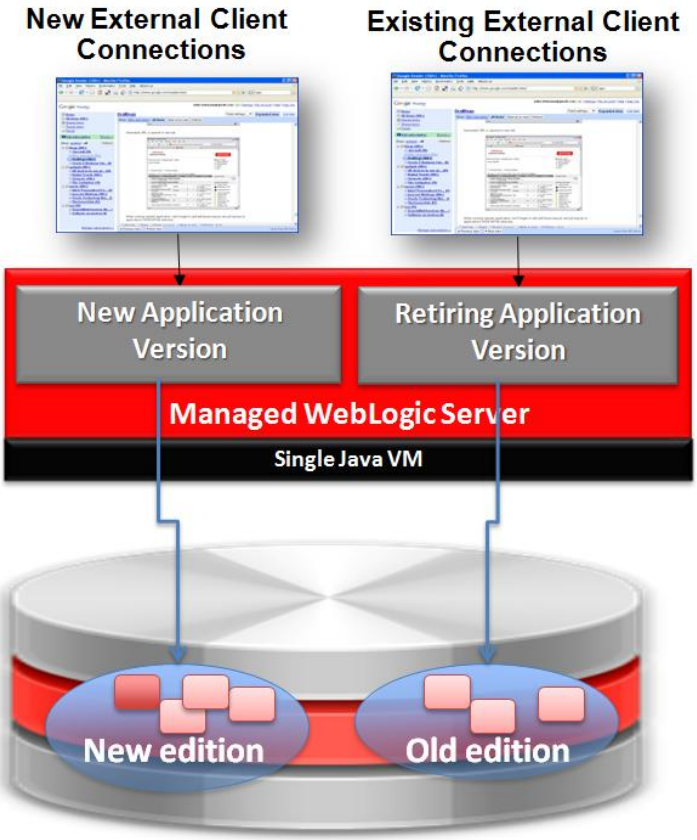


One of the best ways to improve performance in an application, is by not performing certain actions - or at least not perform them more often than strictly required. That helps even more than doing things in the fastest, best tuned way imaginable. The Coherence Grid in WebLogic as well as the Edge Cache can be used to store content that has been retrieved once and reuse that contents multiple times. When the retrieval of that content is expensive in terms of performance - for example because of database query operations, remote resource access or calculations - use of a cache can improve performance and enhance scalability by reducing the load on all involved tier and components of the application, as the data can be reused from the cache rather than retrieved again.

Another way of performance improvement with the WebLogic Platform is through the WebLogic Suite Virtualization option: when running WebLogic Server on the Oracle VM, you can leverage JRockit Virtual Edition and cut out the Guest OS. Performance improvements of 30% have been found compared to WebLogic on a Guest OS in the Virtual Machine, bringing the performance to almost the same level as WebLogic running on a physical OS, without VM.

One final feature in WebLogic that I would like to mention in order to further establish its strength as a cloud platform, is one that deals with one of the oddest concepts in modern application environments that I know of. I speak of the notion of *planned downtime*. We do everything we possibly can to eliminate any unforeseen outage. Availability is strived after in with any possible means, including redundant hardware components, off site fallback environments, on site power generators and clustered software components. Yet many organizations have accepted the fact that when they perform upgrades on the applications running on the platform, downtime cannot be avoided. That is plain silly. Well, it is with Oracle Database 11gR2 Edition Based Redefinition and WebLogic's Production Redeployment. These two features are very similar in their objective and even in their implementation. A short summary of what these features achieve: when the new version of the application is deployed, the current version continues running. When deployment of the new version is complete - both in the

application server as well as at the database schema level - the administrator can decide to have new sessions start making use of the new version. However, existing sessions continue to make use of the now previous version. No one need to experience unavailability of the application - no downtime is enforced. When all sessions that made use of the previous version have disconnected or timed out, this version could be removed from WebLogic as well the database - or it can be left for fast fallback purposes.



DaaS - Development as a Service in the FMW cloud

The Fusion Middleware platform as such does not provide Development as a Service (DaaS). FMW Applications are developed through separate, desktop based development tools. However - as I showed in the previous installment of this column when I discussed the Design Time @ Run Time capabilities in the article *Power to the End User* - many aspects of FMW applications can be configured at run time. This configuration happens through the browser, directly against the run time == cloud based environment. So the run time development of FMW applications is already cloud enabled. In the near future, the range of functionality in applications that can be manipulated at run time will further grow, to encompass most aspects of style, layout, portal- and dashboard composition, static and dynamic content, business logic, communication and process flow. A great challenge is how to test and govern all these run time application manipulations.

Conclusion

Cloud is really a simple concept: increase efficiency and even out fluctuations while ensuring professional management by sharing resources across a large number of consumers. It's a bit like public transportation or communal swimming pools. This article explains these basic concepts of the cloud and discusses various levels of IT cloud services: IaaS, PaaS and SaaS.

Even though at the present, because of Oracle's license policy, there is no public cloud available with a platform powered by Fusion Middleware, the FMW technology stack is very suitable to implement the Cloud platform - as many private cloud implementations demonstrate. WebLogic's abilities to scale flexibly and provide very high levels of availability requiring only limited administration efforts because of integration, consistency and automation of administrative tasks make it an attractive foundation for a Fusion Middleware based cloud platform. Starting today, you can implement a centrally managed FMW platform on virtual infrastructure components that brings most benefits of a public cloud solution to your organization. The larger your organization in terms of IT, the greater your benefits will be. WebLogic Server is the critical component in this Fusion cloud platform - running every Oracle or Java based application, from Form to ADF and from BPM process to WebCenter Space.