Workday’s Technology Platform and Development Processes
Workday’s Technology Platform and Development Processes

Executive Summary

Enterprise applications need to be easy to use and even easier to own. Workday approaches enterprise application design with that guiding principle in mind.

The path to developing new enterprise applications is a revolutionary leap, not an evolutionary process. New enterprise applications require fresh thinking on several levels:

- Delivery model
- Technology platform
- Development processes
- Application design

Workday began as an enterprise software company that made a clean start in all these areas. We adopted SaaS as the best delivery model for enterprise applications and applied innovative design to our Financial and HCM applications.

This white paper focuses on Workday’s technology platform and development processes. It addresses the demands of today’s workforce for enterprise applications. We discuss why a new architectural approach is necessary for enterprise application platforms and why traditional development processes for building and delivering enterprise applications must change. Finally, we describe the components of the technology platform and development processes that are the basis for Workday’s products.

New Requirements

Workday was created with the idea that core enterprise applications can be easier to use and own than they have been historically.

Easier to Use

Traditionally, enterprise applications have not been widely used throughout the enterprise. Back-office professionals were trained to use the application to update and access data. The rest of the enterprise was not active on these systems and depended on these trained professionals and the reports and spreadsheets they generated.

Workday believes that everyone in the organization must be able to use enterprise applications, rather than having to engage IT to perform tasks and generate reports. This self-service approach for employees and managers should not require training or come as add-on modules. We benchmark the overall user experience of an enterprise application against the consumer Internet applications that most workers in any company are accustomed to using today.

Mobile access must be delivered as a native part of the user experience, not as an add-on platform or module or as multiple standalone applications. As user engagement patterns change, the user experience must constantly improve to take advantage of new usage patterns.
Enterprise applications that focus exclusively on transactions usually fail to meet business user expectations. The business user wants to be able to analyze current transactional data in the context of the transaction, and take action based on that analysis. Because traditional systems were built to collect transactions, analysis can only happen in a separate system with separate security, and the user is unable to act on the analysis.

Workday believes that the divides between transactions, analysis, and actionability can be closed with systems that support these three activities natively over a single copy of the data and with a single approach to security.

**Easier to Own**

Enterprise applications must change when the business changes. Businesses have evolved, but enterprise applications have struggled to keep up. Traditional architectures included customization tools with the application so that customers could make ongoing changes to their implementations. Unfortunately, making changes to the applications has become more difficult and expensive. Customers often neglect to update to the latest version because of the complexity involved with many customizations. So, customers cannot take advantage of new innovations and best practices, and they fall behind the pace of the market.

Workday believes that customers must be able to continuously adopt new vendor features without expensive upgrades. For customers to be able to seamlessly apply new system capabilities without costly re-implementation for customizations and data conversion, the vendor must take care of data migration to new releases and updates. When the vendor adopts a continuous-development model where new features are rolled out daily or weekly, vendor responsibility for updates is critical.

Workday has created a new model that allows customers to customize the application without risking the integrity of the system. The traditional way of applying customizations through data-model changes and programming hundreds of lines of code has been replaced with the ability to configure the application through guided interfaces. This model enables customers to continue functioning from release to release because the underlying data model and application code is not modified.

Managing the infrastructure to run an enterprise application is a highly specialized task and requires a team of experts in security, hardware and software scaling, disaster recovery, performance management, system administration, and networking. Assembling, training, and managing such a team should not be required of customers that deploy an enterprise application.

Finally, building and maintaining integrations between enterprise applications and other applications that a customer owns should not be a large part of the overall cost of ownership. Workday believes that it is cost-effective to include integration tooling as a part of the application and to offer more resilient application programming interfaces for integrating systems. If the integration platform is a native part of the underlying application platform, then integrations can surface through the application for easier customer consumption and tighter alignment to the application.
The Need for a New Approach

Delivering against new requirements demands fresh thinking about enterprise application architecture development processes.

A brief review of legacy architectures is helpful. Traditional enterprise applications have employed an architectural approach that can be described as "relational client-server." Developers create a relational model in the database layer to describe the structure of the application. Then, they write code in the business logic layer to store and retrieve data from the database, and to present data and transactional pages in the presentation layer (typically an Internet browser).

This type of traditional architecture allows you to reliably capture transactions at scale and to produce reports of transactional history.

We had three concerns with this architectural approach when we started Workday:

1. Complexity: Complex applications require complex database schemas with many tables to describe application structure and many lines of code to describe the application behavior. Typical enterprise applications end up being millions of lines of code talking to thousands of tables. Any significant change to the application requires making and coordinating changes at both of these levels.

2. Integration: Integration to other systems is accomplished either by getting exports and imports of transactional data (typically in files) or by interacting with an application programming interface (API) at the business logic level. While it is easy to get data out of a relational database, doing this directly circumvents the security and business logic built into the application. Using APIs to interact with the application can be complicated because APIs are typically built after the fact and have to interact with business logic that assumes that it’s talking with a user, not another system.

3. Business Intelligence (BI): These architectures feature detailed transactional reporting, but they do not report in a way that business users care about. Most applications built on these architectures transfer data to separate BI tools to get acceptable reporting and analytics for business users. Once a copy of the data is made, you have to secure access to it and worry about how out-of-sync it is with the live data in the application.

Workday also had concerns with traditional development processes. Traditional development was designed to support large and infrequent releases. Development is done on a separate code line from production. Programs to convert existing customers to the new release are typically built after the release is built. The infrastructure to run the application is either the responsibility of the customer (for on-premise deployments) or an infrastructure team that is not part of development.
These concerns led us to adopt a different architectural approach for Workday’s development and runtime platform. We decided to pursue development processes that would support frequent updates and continuous change.

**New Platform and Development Process**
Because of concerns with traditional relational client-server architectures, Workday decided to take a new approach at every level of its architecture. This approach would fully support the Software as a Service deployment model.

Workday’s mission is to deliver a superior enterprise-application user experience. Workday applications are the center of gravity for everything that is done through Workday’s platform and architecture.

The following diagram features a high-level look at Workday’s platform architecture.

**Workday Applications**
Workday’s foundation is built on Java. We also created a metadata abstraction layer to simplify development. This language is called “XpressO.” It allows application developers to abstract themselves from implementation-specific details that prevent unnecessary technical dependencies and it facilitates Workday’s architecture evolution over time.

Workday application developers can focus on the functional aspects of the applications without worrying about technical and implementation details such as persistence, transactions, scalability, and data center locations. We can focus instead on delivering functional apps with a great user experience.

Figure 3
Different services in Workday’s architecture and platform collaborate to interpret the application metadata that XpressO creates when it processes application transactions and manual or automated requests. By clearly separating how applications are defined from its implementation platform details, new applications can be brought online quickly, as Workday has demonstrated since its early days.

To make applications easy to author and maintain, Workday built its architecture following a core set of design principles that primarily address the challenges outlined in the previous section. These design principles have had a radical positive effect on how Workday delivers applications and our ability to evolve continuously.

One of the important design principles is Workday’s approach to data storage. Workday uses a relational database to store transactional data, but we do not use a relational database to model the core of our applications. All application transactions are persisted in a MySQL database, but the schema for that database bears no resemblance to any of our applications (for example, there is no Account or Worker table). The schema consists of just a few tables optimized for storing application data and metadata.

To maximize security, all access to the database comes through Workday’s application services. This setup means that no customer employees have direct access to the database (not even IT). Using a simplified schema to store transactional data means that we do not have to worry about schema changes for all of our customers when we release new features and perform weekly upgrades.
A second very important design principle is the use of a metadata object model to describe the structure and behavior of the application. Here is where XpressO comes into play. There is no code-based procedural logic in a Workday application. Instead, Workday developers define the structure of our applications by defining classes for the key business objects in the application. Classes can have relationships to other classes, attributes, and methods. Methods define the behavior of the application, and the business logic is defined by declarative relationships without the need to write any procedural code.

All of these parts of the object model (classes, relationships, attributes, and methods) are created through a forms-based set of tasks. The resulting application is a collection of metadata definitions for each part of the application object model. These definitions are stored as collections of simple Java objects in the memory of the Java virtual machine (VM) that is the runtime for all Workday applications. The Java runtime knows how to interpret the metadata definitions into the transactions (tasks and reports) that make up the application.

Based on these two design principles, Workday decided to maintain core application data in memory, along with all of the metadata definitions of the application. Having the data in memory means that most Workday reports are produced without having to access the database. Having the in-memory data organized in an object model that features a rich network of relationships between classes means that Workday reports can offer multidimensional analysis along with data presentation.

A simple example of this capability is a headcount report. To display the headcount for an organization, the application collects and counts all instances of all workers in that organization. The result is an in-memory cube where you can further explore all the dimensions of data relating to the worker object.

Figure 5
This approach to defining and running applications has many benefits:

- **Developer productivity**: You can think of relational client-server applications as millions of lines of code interacting with thousands of relational tables. The definitional approach that Workday uses results in applications that are millions of cross-referenced metadata definitions. While there is nothing simple about huge numbers of metadata definitions, organizing the application this way enables our developers to continuously enhance and refactor our applications and to deliver these changes into existing customer implementations in an upgrade-safe manner.

- **Single transaction processing model for requests from users and systems**: All requests coming into Workday are processed the same way. The request must come from a valid and authenticated user ID. All requests are put into the same queue for processing and are processed and audited the same way. There is no separate processing approach or language for “batch” processing vs. online processing, and there is no direct read or write access to the relational database.

- **Single security model for all data access**: Role-based security is used for all transactional access to data in Workday applications. Complex systems often have different ways of securing access to data. This is especially true of access from custom reports and analytical tools. In Workday, the custom report writer applies role-based security at a granular level of data for all reports that a customer creates. The same role-based authorization applies to mobile access, analytical tools, dashboards, the Insight Service, access through our web services API, and workflow processed in our Business Process Framework.

- **Configurability of applications**: Workday supports configurability and extensibility of its applications. Customers create their own metadata for assets like business processes and custom reports, and they add their own custom fields to existing Workday objects. It is difficult to make sure that code-based extensions to an application are “upgrade-safe,” so this configuration is done without coding. The result is a tenant-specific version of application metadata that Workday’s runtime knows how to process and update, even for a new release. Workday’s customers can configure with confidence knowing that they will not have to revisit and repair their work on successive updates to the product.

**Technology**

The Technology services involve the most innovations and evolution over time. Each tenant’s application requests are processed via a collection of specialized services that execute the requested transaction in a secure and scalable manner that guarantees optimum performance. This capability evolved from a unique central processing service to a collection of well-coordinated distributed services.

The Technology services can be subsumed into three main sets: Transactions, Data Management, and Compute services.

**Transactions**

The Transactions service is primarily responsible for processing requests coming from user-triggered actions, such as tasks or reports. The Transactions services use an in-memory representation of the tenant’s data and execute the needed request logic. Each tenant has a single “update transaction service” (object transaction service or OTS) that manages all transactions that update any information.
When a task request comes in to Workday, it's handed off to the appropriate class in the object model that knows how to process that update. The appropriate methods are called to process validations on the updated data. If the validations pass, the updates are made to in-memory data and are recorded in the database via the Persistence services. The application must receive a commit from the database update for the new values of the in-memory data to be visible to other transactions in all active tenant-compute services.

The database is always the system of record for all application data, even though it is rarely used for reporting and querying, thanks to the in-memory copy of the data. The particular aspects of synchronizing and managing data in the Transactions Service and Compute Services are performed by the Data Management services.

Data Management

The Data Management Service is at the core of Workday’s scalability approach. One of the primary components of the Data Management Service is the “data cache” that maintains a compressed copy of all core application data and metadata (or “instance cache”) comprising the Workday application. Workday uses the instance cache to quickly make snapshots of customer data when starting up one of the report or calculation-compute services mentioned earlier, and to reconstruct data in the memory of the customer’s Update Transaction Service or other Compute services to a specific time. This allows the instance cache to be selective about which instances must be in the “main” Update Transaction Service memory without having to pay the price of going to disk if we need to access a less frequently used instance.
Video Storage, Streaming, and Interactions

Workday Media Cloud provides video upload, storage, and streaming services to a variety of Workday applications. The service is used extensively by Workday Learning and is central to providing an up-to-date, employee-friendly learning experience that allows anyone to share and consume training through video.

Under the hood, Workday Media Cloud employs a hybrid-cloud strategy in deploying to both Workday data centers for persistent metadata storage and utilizes Amazon Web Services for video handling to provide direct upload, elastic compute, elastic transcoding, and scalable storage. In addition, the service leverages Akamai to provide global content delivery, ensuring that high-quality video content can be accessed and streamed anywhere in the world. Workday Media Cloud has also built its own playback technology that enables dynamic quality switching (HLS).

Figure 7
Compute

Not all processing happens in the customer’s Update Transaction Service. Workday’s Compute services allow us to offload requests for reports, searches, printing, integration logic, and large calculations (such as Payroll) to dedicated services running on separate Java virtual machines on separate hardware to scale appropriately. This process enables Workday to distribute processing more efficiently and effectively when greater compute cycles are needed for longer-running requests or increased capacity.

A set of Big Data services also analyzes third-party data loaded by customers into Workday’s cloud. These services return the results of MapReduce operations over the customer’s data, which is stored in Hadoop, to the customer’s Data Management services for use in Workday reports via the Insight Service. The Insight Service also provides scores such as retention risk, which are the result of machine learning models that access data through Workday’s SYMAN (SYstematic MAss Normalization) service. SYMAN’s service manages the collection and analysis of data, and it normalizes terminology and applies machine learning to generate predictions and recommendations for the Insight Compute service.

Machine Learning/Artificial Intelligence

Workday Machine Learning/Artificial Intelligence (ML/AI) includes a broad set of technology initiatives for delivering automation, predictions, and recommendations to customers in order to enable them with faster and smarter business decisions. In 2014, Workday introduced our first applications that leverage data science and machine learning. Today, the scope of our Workday investments has expanded significantly. These investments can be categorized into three primary areas: applications, ML platform services, and infrastructure.

Workday embeds ML capabilities natively into Workday applications for intelligent experiences. These applications fulfill specific use cases for our customers in the functional areas of financials and HR. As an example, our Workday Learning application features ML-powered search, allowing users to more efficiently find relevant training materials. It also features course recommendations, taking into account content relevance, popularity, and ranking, so learners can continue finding and engaging with more-relevant training content. Workday also offers ML applications to address specific business requirements. As an example, an application called Retention Risk is designed to predict a worker’s risk of leaving the company.
From this view, customers can see associated retention risk levels for each employee, for each organization, and for the entire company. These risk predictions are produced from a machine learning algorithm that considers a wide number of factors such as compensation, time in current role, distance to work, performance, and many others. Identifying these risks allows HR partners to proactively engage with and make steps to retain high-performance workers.

ML services are enabled by the ML platform, which features a multitier architecture. By building ML capabilities as services on this platform, features and algorithms can be repurposed and uptaken into many different applications, both existing and in the future. As an example of these services, the Workday time series forecasting algorithm has recently been integrated into Workday Planning.

The function can analyze a dataset with a time series, and based on patterns in the data, produce a predictive trendline that accounts for seasonality. There are many ways this function can then be used. For example, a retail customer could use this function to make more-accurate predictions about seasonal contingent workers needed in their workforce planning. They could also then use it for financial planning, in attempting to predict what their seasonal product revenue lift will be around December holidays.

Lastly, at an infrastructure level, Workday is regularly evaluating state-of-the-art technologies to support the use of machine learning. We consistently introduce new components to enhance computing capability with scale and performance to process data with significantly increased volume, variety, and velocity. Workday also invests heavily in data management, including data access, persistence, and retention.
In-Memory Computing

Workday has always been an in-memory application. From day one, Workday has kept core customer data in the memory space of our Java-based OTS. However, over the years, Workday has evolved its single OTS into a separate layer of services (Transactions, Data Management, and Compute services) to take advantage of the benefits of in-memory data in order to bring higher levels of performance and scalability to our applications.

Workday believes that continuously enhanced in-memory data management is the path to continuous scale for our applications. It is no longer a problem to be solely addressed by databases on disk, serving data to applications via SQL. Our in-memory data services are an essential part of a highly optimized multi-level caching approach to managing application data.

Figure 9 illustrates how the Update Transaction service interacts with the Data Management service to retrieve data for processing on the Update Transaction service and other tenant live-compute services.

In Figure 9, the Update Transaction service (the same mechanism that works for the Compute services) started from the Instance Cache are kept up-to-date with “change sets” corresponding to ongoing transactions issued from the tenant’s Update Transaction service (Step 1 in Figure 9). Once a transaction is updated to the database (Step 2), a change set is queued for consumption by the Instance Cache and all other related live-tenant Compute services (Step 3). Before performing a requested report or calculation process, any of the live-tenant Compute services will check a version flag in the database to make sure that it has the latest updates. If it is not up-to-date, all queued change sets are applied before performing the report or calculation. This process guarantees that the information used by any of the services in the Compute services is current up to the most recent transaction (Step 4).

While the Instance Cache safeguards the access of transactional data and metadata, the DocStore service manages the storage and retrieval of unstructured data stored in a NoSQL data store. There is more information on this process in the Persistence layer section of this paper.
For classes with many millions of instances, Workday maintains a Query Service that works in combination with the Instance, which allows us to do reporting with drill-down to multiple dimensions over these large datasets. This service is used heavily for reporting over high-volume objects, such as financial journal lines. It relies on an in-memory Search Index for performance.

Last, the Search service allows searching for terms contained within Workday data and on a set of normalized search terms in SYMAN to enhance the relevance of results for a given search request. For instance, look up “Jon” to find workers named “Jon, Jones.”

Platform

Platform services introduce a set of unique reusable and generic components to be used within the Technology layer. Among the services found in this layer, we find functionality such as authentication and authorization, communication, logging, monitoring, and customer landscape information. Regardless of the request coming from a report, task in a browser or mobile client, or integration, the same service is always relying on a single source of truth. This process enforces application behavior consistency and also lowers management efforts and costs.

Service Discovery allows Workday to understand each tenant’s available distributed compute services. As described in the Technology section, each customer always has Update Transaction services, and as more computing power is needed, additional Compute services are launched to respond to increased customer demand and provide optimal performance.
When a new tenant request comes in, Workday interacts with this service and determines if it is possible to route it to an existing Compute service or if it needs to dynamically launch a new one. This, in a nutshell, is how Workday’s infrastructure dynamically scales out as demand increases.

The Auth service centrally manages security. This service helps enforce and secure access to data across all Workday applications. This process contrasts with systems that have been put together under a single sign-on umbrella but still require several authorization frameworks for different modules in a business suite.

The logging and monitoring services record the action of every transaction so that it is always possible to audit “who did what and when.”

**Persistence**

The services in the persistence layer manage access to data used by Workday applications that requires a level of persistence and time durability. Usually the data management set of services in the technology layer directly interacts with the services in this architecture layer.

Workday does not rely exclusively on a relational technology for persistence as it once did. While MySQL is used for transactional data, unstructured data (such as resumes, business process documentation, and photographs) are stored in a NoSQL database. Workday uses NoSQL for its capacity to scale to very large data volumes and to replicate.

Workday’s platform also allows customers to upload large external datasets to our cloud, which is stored in a Hadoop file system. Workday selected Hadoop for its ability to accommodate large volumes of data and for its flexibility in handling data in any form. When external data is correlated with data from Workday’s applications the customer gets superior analysis and reporting.
There are benefits to Workday’s approach to storage:

- Using a simplified schema to store transactional data means that we do not have to manage schema changes for all our customers when we release new features and perform weekly upgrades.
- Using Hadoop’s schema-less approach makes it easier for customers to load their data into our cloud.

By not tying ourselves too closely to any storage standard (such as SQL) we are able to select the best storage technology for particular storage needs.

As an example of these benefits, we changed the way all transactions are stored, from inserts of multiple rows relating to business object changes to combining all changes in a BLOB data type stored with a single insert. This optimization reduces the size on disk by more than half. It also halves the time it takes to initially instantiate data from disk into memory. This optimization would not have been possible if we were performing SQL updates against a complex schema.

Not being tied to one standard also keeps Workday’s options open for switching to emerging storage technologies without disrupting our customers or our application developers. With innovation in storage technologies at an all-time high, this storage architecture ensures that Workday is continuously able to leverage the best current approaches to storage. Martin Fowler, an industry thought leader on object-oriented design, refers to this approach as “Polyglot Persistence.”

**Infrastructure**

The infrastructure layer of Workday’s architecture deals with the typical aspects found on IaaS services. To summarize its function in the overall architecture, this layer is essentially Workday’s operating system. Automation helps multiple services-provision servers install the appropriate services on customer-specific configurations, and make them available to a tenant as part of instantiated compute services. Without a full end-to-end, automated infrastructure-provisioning system, it would be impossible to scale dynamically at any time to support current business demands.
When we designed and built Workday’s infrastructure, Workday adopted a “scale out” architectural philosophy. We increase storage capacity and enhance performance by adding additional servers, storage, and networking, rather than by vertically upgrading to larger models or appliances. Similarly, for our software architecture, Workday scales application capacity through additional services running on our infrastructure.

Scaling horizontally provides a number of benefits as they relate to service availability. First, the servers are redundant. Workday’s applications will not go offline if one of our servers fails. Secondly, it is possible to increase storage capacity without sacrificing performance. Workday customers are assured that, as the infrastructure layer scales to handle more data, they will not have application performance issues.

**Integration and User Experience**

Two other important aspects of Workday have direct correlation to Workday’s architecture: Integration and User Experience. These have not come as an afterthought and have always been native components of the platform.

**Integration Services**

Workday provides a lot of functionality with its business applications, but customers often need to integrate with third-party applications. For example, a very common integration scenario in most of Workday implementations is that Workday Human Capital Management integrates with benefit providers. Integration with external systems has never been easy. However, Workday provides multiple alternatives to simplify connectivity to other systems.

![Diagram of Workday Cloud and Integration Cloud](image-url)
Workday believes that web services are the best approach for integrating applications. Workday applications are built to automatically generate web services for all tasks, allowing outbound and inbound communication with other systems. However, as there is no single, agreed-upon standard for how web services are designed, Workday embeds enterprise service bus (ESB) technology in our platform. These integration services flexibly transform inbound and outbound data payloads and vary the delivery protocol to meet the technical specifications of third-party applications.

Integration services also enable our customers to build their own integrations and to execute them in Workday’s cloud. Workday accomplishes these integrations by offering two tools that interact with the Workday Web Services Application Programming Interface (API): Enterprise Interface Builder (EIB), and Workday Studio.

- **Enterprise Interface Builder (EIB):** EIB allows users to create simple, one-way in or one-way out integrations using a forms-based user interface. Once an “integration system” is built with this tool, the customer can schedule execution of the integration in Workday’s cloud.

- **Workday Studio:** Workday Studio is a powerful tool that allows customers to build complex integrations between Workday and one or more third-party systems. The end result of Workday Studio is the same as with EIB: an integration system that can be scheduled to execute in Workday’s cloud.

These tools are included in the purchase of any Workday application at no additional charge. Customers can always interact directly with Workday’s public web services API with the programming language of their choice. Workday offers an open, standards-based API for programmatic access to our business services. We provide three types of API:

- **Public web services API:** Workday offers an open, standards-based API for programmatic access to business services within Workday.

- **Reports as a Service (RaaS):** Workday enables customers to build their own web services when they create custom reports through our Report Writer. Custom reports can be web service enabled, exposing the report as both a SOAP- and REST-based web service for integration.

- **REST API:** The Workday REST API is targeted for applications that do small, typically self-service, transactions initiated by users. Because a user initiates the interaction, the REST API is designed to quickly return a small set of frequently-used information for display or further action.

There are several benefits to including integration services as part of an application platform:

- The use of ESB technology to transform data payloads and select the appropriate delivery protocol makes Workday look like a more flexible endpoint to all of the applications it connects to.

- Including tools with Workday applications allows customers to decide which integrations they want to run in our cloud. When customers leverage our cloud, they lower the cost of building and maintaining integrations that run on their infrastructure.

- Integrations can be surfaced within the application and seamlessly interact with a business process, making integrations context-aware and configurable.
User Experience (UX)
Workday entirely separates the generation of its user experience from the definition of its applications. Application developers do not do pixel-perfect screen layout, and they never specify where a given field appears on a page. Instead, application developers define the fields involved in a transaction. They specify how the fields are grouped and how they are ordered. User Experience Services generates the presentation of the transaction from this information.

There are several user experience benefits to this approach. A generated UX provides consistency. Workday does not have to teach and coordinate hundreds of developers to lay out their pages in the same way.

Separation of the UX from the definition of the application allows Workday to change UX technologies without changing our application.

Workday’s browser presentation layer is currently HTML5. This is the third browser UI technology that Workday has embraced (after DHTML and Flash). Workday believes that UI technology will continue to change faster than applications change. So, our architecture allows us to stay current by changing our presentation frequently. The mobile experience has taught us that the UX gets stale if its design is not refreshed every other year, at a minimum. The same is true for the browser.
Mobile

Mobile access is a foundational feature of modern applications, not a separate product or platform. With Workday, mobile access is simply another aspect of User Experience Services and is included as a part of every application.

In addition to desktop-browser access, Workday applications can be used on a native iPhone, iPad, or Android client, or any modern mobile browser. Each client’s user experience is optimized based on touch and form factors, so the user never has to pinch and zoom to find fields or buttons meant for the real estate of a desktop browser. Native mobile applications offer optimal usability. Workday passionately supports native clients in conjunction with making our HTML5 browser client completely responsive to the tablet and phone feature set and form factors.

Because Workday’s mobile solution is simply an extension of the application, mobile users are accessing the same data, business logic, and functionality as they do on a desktop. For example, a dashboard built on the desktop can be accessed and modified on any mobile device, with no data persisted on the device. What’s more, Workday has designed its mobile clients to utilize the same single security model that customers establish for browser access.

This allows customers to easily implement Workday’s mobile solutions. The only setup task (other than downloading the app) is to grant all appropriate users the ability to use mobile devices to access Workday.

Worksheets

Spreadsheets have had and will have continued relevancy to the business due to their unique capabilities to construct insights through modeling, analysis, and visualization. Spreadsheets continue to be the number-one business intelligence tool as measured by breadth of distribution and daily usage.

The problem customers have today is that their spreadsheets are disconnected from Workday, resulting in a lack of trust in the validity of the data and the derived answers: the data they contain is not live, up-to-date information that reflects the state of the business. Additionally, the data is insecure, often residing as files on end-user machines outside the security and audit controls provided in the Workday technology platform. Where customers tried to deploy non-spreadsheet-based BI tools, they found challenges with wide-scale adoption as the tools were not intuitive to users grounded in the spreadsheet UX.

Worksheets is an end-user analytics tool that provides the usability and flexibility of a collaborative, enterprise-class spreadsheet with the power of the Workday technology platform. Worksheets allows end users (employees, managers, and executives) to collaborate with secure, “live,” refreshable Workday data as well as external data, with real-time user presence, secure data sharing, and engaging visualizations via the web or mobile devices.

Worksheets addresses the problems that exist with the current spreadsheet model: it provides a single source of truth by leveraging live data extracted from (and linked to) Workday report data, uses the singular Workday security model for access and authorization, and provides an intuitive user interface enabling users to move in context from a spreadsheet into detailed records.

Working with live data is essential. Reducing the friction of getting and refreshing data in the spreadsheet is a key productivity enabler for our users, providing confidence in the timeliness and accuracy of the data and having context with the ability to drill right back to the source data.
As a microservice embedded seamlessly within the Workday cloud, worksheets benefits from the core advantages described above from a development, deployment, and life-cycle perspective. Customers benefit from the single security model that provides worksheets live data access. Users access and share data based on their security access. All actions, including sharing and collaboration within the worksheet, are captured.

The worksheets microservice includes a Java-based sheets engine and sheets web component that provide the compute, connection, and persistence capabilities needed to power the spreadsheet. All computation is done on the server, enabling the Workday cloud to provide support for large-scale spreadsheets independent of the user’s client device capabilities. Communication to the client is via REST and WebSockets to enable high-performance, real-time updates—a requirement for simultaneous co-authoring of the workbooks.

The worksheets microservice supports high availability across its underlying components, and is designed to scale out to meet customer workloads.

Customers can rely on worksheets for answers to complex ad hoc analytics questions, collaborative data modeling, or simple data manipulation—from anywhere and while on the move. Worksheets is available as part of the Workday HTML5-based web interface, as well as on a native iPhone, iPad, or Android client.

Worksheets facilitates adoption by providing high-fidelity import of customers’ existing Excel spreadsheets. Our extensive formula library provides significant parity with Excel and extends beyond those capabilities to enable enterprise data modeling and complex analyses. Additional user value on top of traditional capabilities includes coherent use of core Workday technology such as notifications at both a process and functional level as well as functions specific to enterprise data modeling.

### Development Processes

Workday is committed to delivering frequent updates while keeping our customers current on the latest version. This commitment influenced many of Workday’s design choices for our platform architecture. It also led us to reconsider development processes as they relate to both products and platform as well as an infrastructure.

Workday embraces continuous deployment of new product and platform functionality on a single code line. This approach is the only way to support ongoing delivery of many new features to our growing customer base. Workday developers work in small scrum teams. These teams work in an agile manner. They own design, development, and quality, and they iterate to create features ready for production.

Once a feature has passed through the test pipeline, it is checked into the production code line. Software switches called toggles are used to determine whether each feature is viewable only to development, to customers in a preview mode, or to customers in production. Supporting our ability to work on a single code line is a large, automated test infrastructure.

Workday has also developed a process for converting customer data in the background as new features are checked in. This approach dramatically changes the work our environments and infrastructure teams perform on the day of an update. Instead of moving to a new code line and running conversion processes for all customers on a single day, an update involves simply switching the new features from preview mode to production.
A second benefit of organizing around scrum teams is increased cradle-to-grave ownership of services. Our scrum teams own the development, deployment, and monitoring of identifiable runtime services. This enhances Workday’s overall service delivery and our responsiveness to our customers.

Workday believes that our datacenter infrastructure is just another system that needs to be continuously enhanced as we expand our products and our customer base. For that reason, infrastructure is a part of the development organization and DevOps projects are a part of every new Workday update. Increasingly, technology makes it possible to treat the datacenter as another software resource.

Workday makes an ongoing investment to increase automation of services and virtualization of compute and networking services to more efficiently allocate and share infrastructure resources for a growing set of customers.

**Conclusion: Embracing Continuous Change**

Workday took advantage of being able to start with a “clean sheet of paper” at its founding in 2005. We took a fresh approach to technology in response to unmet requirements for enterprise applications. The early decisions we embraced were:

- Object-oriented (as opposed to relational) design of applications
- Definitional, metadata development (as opposed to coding)
- Application data available in memory
- Generated user experience

These guidelines have allowed Workday developers to focus on business logic development without having to be concerned about in-memory data management, persistence, or UI technology. The business logic itself is not tied to a specific underlying technology (such as SQL, HTML, or JavaScript). This cleanly separates the business logic that defines our applications from the technology platform that runs them and allows us to develop the applications and the technology platform independently, without breaking one another.

When Workday was launched, our architecture could be described with this simple picture:
Over the years, Workday has stayed true to its early architectural decision while continuously evolving the implementation of our architecture. Today’s picture is a bit more sophisticated:

Figure 18

Workday firmly believes that to deliver applications that are continuously relevant and modern for our customers, it must embrace continuous change at all levels. Workday is able to support the ongoing configurations and customizations that our customers make to their implementations of our products. Workday is also able to continuously deliver new functionality that customers can adopt without expensive upgrade projects. And, while all this is happening, Workday is able to continuously innovate, refreshing the implementations of our architecture.

Our belief is that enterprise applications need to become more like consumer Internet sites, such as Facebook, Amazon, or Netflix. These sites are sophisticated applications that are updated continuously. Customers of these sites are not aware of the details of the infrastructure, but they expect a beautiful end product. Similarly, customers do not need to know what version of the site they are on because there is only the current version.

Workday’s architecture and development processes are designed to make the experience of using our applications similar to using one of the big consumer internet applications. Workday is committed to being able to continuously change our applications and our platform, to support our customers while they achieve their business goals.