AppStack Technology Overview

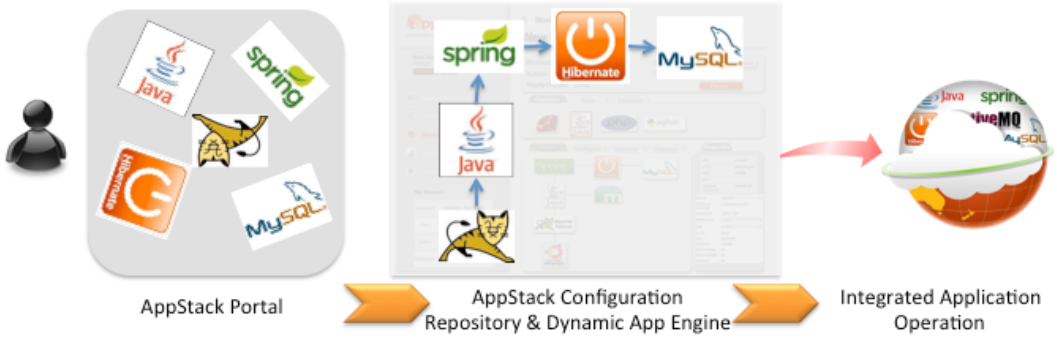
Model-Driven Application Management for the Cloud

Accelerating Application Time-to-Market

The last several years have seen a rapid adoption for public and private cloud *infrastructure* services (IaaS). The need for instant and on-demand server provisioning was the main driving force behind the massive growth. Public cloud services such as Amazon Web Services EC2, The Rackspace Cloud and others have become primary targets for application developers who need the rapid build up and tear down provided by these environments to enable rapid development, test and QA of new applications. The agility provided by public cloud infrastructures is also becoming available for enterprise "behind-the-firewall" environments, and is enabled by enterprise cloud orchestration layers such as Citrix CloudStack, CA Applogic, VMware vCloud Director and others.

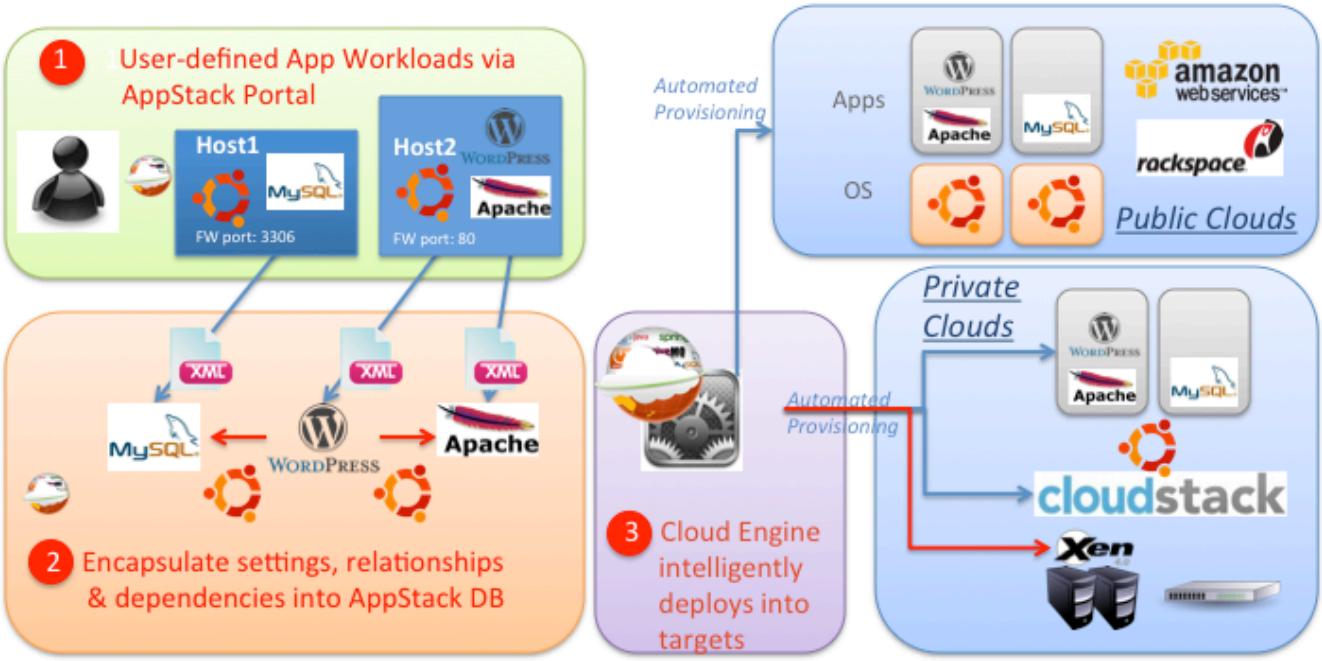
Given the acceleration provided by these IaaS clouds for rapid infrastructure provisioning, it is only natural that enterprise customers look for solutions that can accelerate application services in these cloud environments. A group of vendors have provided on-ramp solutions through server templates and script automation services. These products have further simplified application installation for simple and pre-defined application stacks, but still require extensive scripting and technical knowledge to use effectively. But enterprise app adoption in the cloud is still rare due to the lack of holistic management capabilities, especially for more complex applications as found in the enterprise. PaaS on the other hand provides integrated application development capabilities, but only on public cloud infrastructures, but only with fixed, rigid and non-extensible application stacks.

Appcara takes a very different approach to facilitating and managing cloud applications. With a goal of completely automating cloud application deployment, AppStack uses a *model-driven approach* to capture and assemble user-defined apps in real-time, including the relationships between multiple servers and all of their configuration data. The AppStack database model, termed the Configuration Repository, is then used to dynamically track and reflect any changes in the relationships and dependencies of the application, and to do so automatically without user intervention. AppStack provides an end-user friendly centralized management Portal to truly simplify the process of defining complex applications as Workloads, provisioning them into multiple cloud targets, and managing the application lifecycle seamlessly. AppStack utilizes a powerful set of logic and workflows, termed the AppStack Dynamic Application Engine, to provide the automation layer, with an ability to provision applications consisting of a mix of components developed from different sources, in different languages, libraries and platforms. On top of that, AppStack provides complete application portability over both public and private infrastructure clouds. This allows applications to be deployed as a "one-click" operation into a choice of public or private cloud targets. The flow is depicted below.



The AppStack Architecture

AppStack was designed to support the highest levels of user simplicity for cloud application management, through extensive automation, dependency management and abstraction of cloud and infrastructure level details. The system is comprised of three main patent-pending components: the AppStack Portal (1), AppStack Configuration Repository (2) and the AppStack Dynamic Application Engine (3), as referenced in the figure below.



The AppStack Configuration Repository provides the scalable repository to capture all application related information, including all components, property settings and the relationships and dependencies between the various components. The key aspect of the AppStack Configuration Repository is its dynamic nature, as it is continually updated to reflect changes in the application properties or relationships, in order to accurately reflect its stage in the application lifecycle, or its deployment environment. The AppStack Configuration Repository is designed in a manner so that infrastructure and operating system level dependencies are de-coupled, to make application Workloads portable across lifecycle stages or cloud environments.

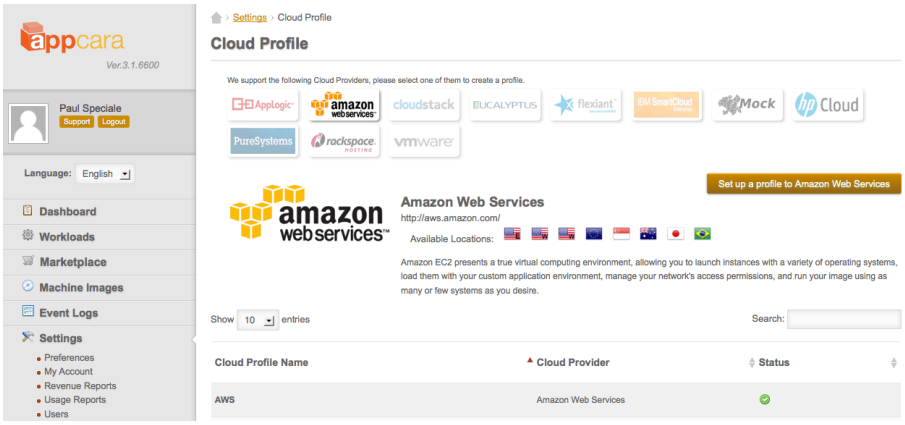
The Dynamic Application Engine provides the automation intelligence for the environment. The Dynamic Application Engine interacts with the Configuration Repository for reflecting changes in application state or dependencies, when new application Workloads are defined, or when Workloads are provisioned into a new stage of the lifecycle or a new cloud environment. The key aspect of the Dynamic Application Engine is its ability to automate all aspects of lifecycle management, externalized as one-click cloning operations, and its ability to provision Workloads into a wide variety of cloud targets both public and private. The Dynamic Application Engine encapsulates cloud specific logic and utilizes it in conjunction with the AppStack Configuration Repository model to make provisioning totally transparent and fully automated.

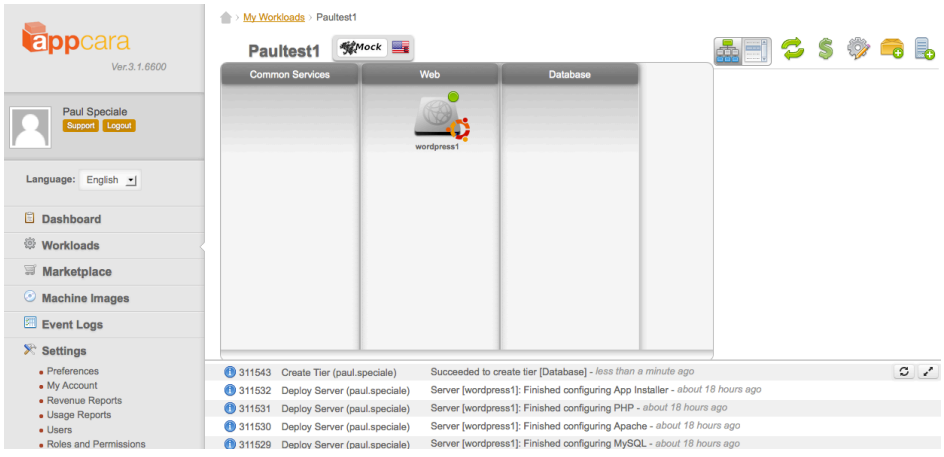
The AppStack UI Portal

The AppStack portal provides a user-friendly web-based interface for technical and non-technical users alike, to define, provision and manage cloud applications. A typical Portal user would be a development team's project manager, release manager or even the developers themselves, enabling self-service provisioning of full development stacks, test & QA, or production runtime environments.

Through the AppStack portal, users can browse existing Workloads, define new Workloads, Provision Workloads into a variety of cloud targets, and manage previously provisioned Workload instances. The Portal provides a series of pages for Workload definition and editing, provisioning, lifecycle management and monitoring. The Portal provides an extensive built-in catalog of popular pre-defined applications, including Wordpress, SVN, Redmine, Hudson, Ruby, Apache, memcached, MySQL, MongoDB, MySQL and others. In addition, users can import their own custom applications into the system and freely mix them into complex Workloads with other custom or pre-defined application stacks. The ability to customize and mix applications provides a key AppStack differentiation from the fixed application stacks provided in today's PaaS offerings.

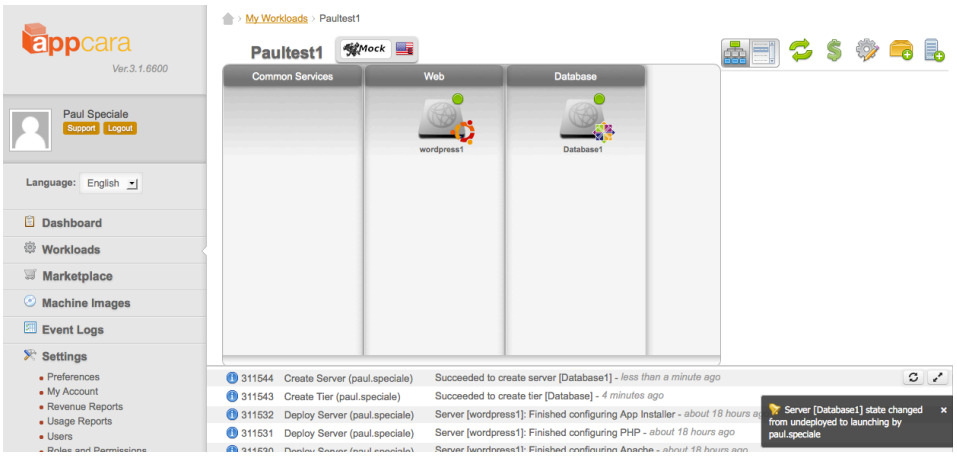
The Portal provides the user with easy to use pages for configuring credentials within Amazon Web Services EC2, and The Rackspace Cloud, as depicted below. Users specify logon credentials including all required security keys for each environment, to allow AppStack to automate all application provisioning. Private infrastructure clouds such as Citrix CloudStack and CA Applogic are supported as well, with full multi-tenancy support to make it possible to deploy in Service Provider, or multi-departmental deployments within the Enterprise.



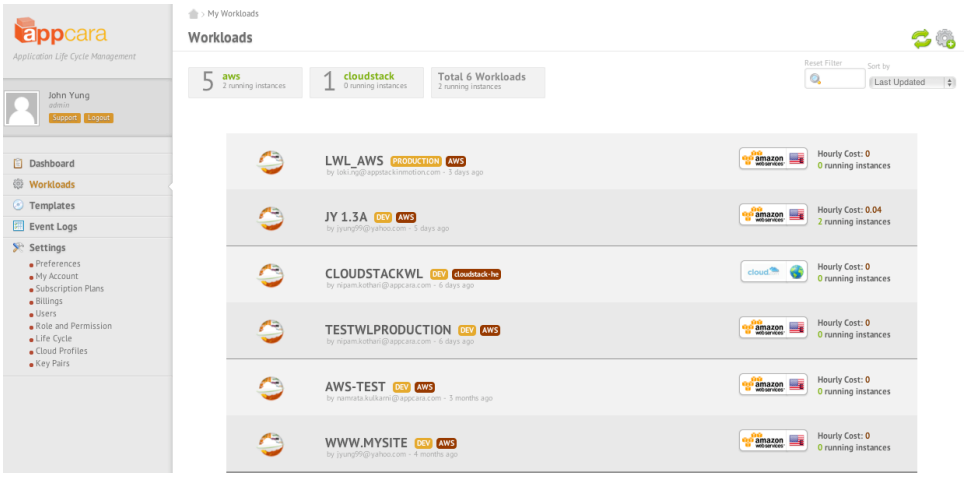


Defining new Workloads is enabled by the Portal's Workload wizard and Topology designer, which provide simple point-and-click interfaces for the user to add and configure application elements to the Workload. Workloads can consist of top-level applications, such as a Wordpress application to enable new website services.

In general, applications can be defined with any number of servers, organized and visualized in a multi-tier deployment model. The UI Portal enables all Workload definitions without the need for scripting. Furthermore, the system allows the user to edit application properties directly in the UI, for example to set specific server port numbers, or database ID strings. This eliminates the need to step out of the environment to directly edit configuration files, thereby maintaining a single-pane-of-glass management interface. Once a Workload has been defined, it is ready to be provisioned into any of the cloud environments previously set up by the user. Provisioning is a simple one-click operation, with the Dynamic Application Engine automating all aspects of the process, and informing the user of status via the Portal, or more complex multi-application Workloads.



The Portal provides a list-view for all existing Workloads, their deployment status, and their target environments. AppStack also reports the aggregate cost of a Workload in each public cloud environment, directly accessed from the underlying AWS or Rackspace clouds.



Dependency management also plays a critical role in automating lifecycle management, for example to move applications from development to testing, or testing to QA phases. Developers may desire to unit test their applications in a variety of environments, and QA may desire to performance or stress test applications under higher loads. All of these parameters create requirements for unique application environments to provision the application into. This has historically been a very manual, and error-prone process, for developers and testers alike. With the AppStack Portal, this process becomes automated. Simply put, AppStack users can perform single-click Workload-level cloning operations through the Portal to dynamically create additional application environments for testing, QA or even production Workloads.

As a simple example of dependency management, a user creating a Wordpress Workload would be alerted to the dependency that an Apache server is required for this application. The Apache server can be added with a single-click, and its properties can be directly edited through the Portal. In general, AppStack auto-detects all required libraries and applications. The system furthermore detects remote servers and can automatically manage these dependencies through configurable Advanced Actions on these links, including “follow new” or “keep existing” host policies. These policies are critical to automating lifecycle management and eliminating common configuration errors that can cause developers to waste hours to track down, debug and resolve.

AppStack Configuration Repository

The AppStack Configuration Repository provides the model representation of all Workloads managed by AppStack. The AppStack Configuration Repository is a dynamic, extensible database purpose-built to model all aspects of complex Enterprise application Workloads.

As described earlier, the AppStack Configuration Repository is a scalable meta-repository that captures all Workload related components and relationships. The key aspects of the Configuration Repository are to capture and track all components (component level applications such as Wordpress or MySQL), their configuration properties and their dependencies. In order to provide agility and respond to user changes, the Repository is continually and dynamically updated to reflect changes in the Workload such as property changes, addition or deletion of components, or changes in dependencies. The AppStack Configuration Repository is designed to abstract away all infrastructure and operating system level dependencies to make application Workloads independent of the underlying Infrastructure cloud, the operating system, instance sizes and hence create portability across lifecycle stages and even across cloud environments. Moving a Workload from a public to private cloud can now be facilitated with a single click operation.

More information

For more information, please contact Appcara on the web at www.appcara.com, or email us at info@appcara.com.

AppStack Dynamic Application Engine

The AppStack Dynamic Application Engine was designed to work in conjunction with the AppStack Configuration Repository to provide full automation of cloud application management. The Dynamic Application Engine itself consists of several sub-component managers that enable the core capabilities such as Workload Manager, Lifecycle Manager, Mobility Manager, & Application Manager. The Dynamic Application Engine provides an API for tools and applications to access the core services, and the AppStack Portal in fact interfaces through the API. The core Application Engine is fully multi-tenancy enabled, with an internal repository for managing multiple accounts and users. The system abstracts external Cloud Services (AWS, Rackspace and private clouds such as CloudStack) to simplify the interface layer to these services, and enable portability. The Dynamic Application Engine utilizes Appcara's specific IP to decouple applications from their non-portable OS layers, to enable simplified and portable provisioning across cloud services.

Portability Layer

AppStack has been designed to provide portability across cloud environments. The system architecture provides an abstraction layer that isolates all low-level dependencies of each cloud environment. For example, cloud environments differ in their definitions of virtual server sizes (e.g., Amazon Large Instances), local storage, and the availability of common shared services (load balancers, for example). The system also decouples applications from their operating system dependencies, thereby providing a higher degree of agility and portability. This allows AppStack to automatically provision applications with little to no modification, across all supported clouds, and enables application portability across clouds. This feature of the AppStack environment helps eliminate environment lock in and allow users to deploy applications across the right environment to fit the business need.