Manage application services with virtual application patterns

Virtual application patterns capture years of cloud application services management expertise

Bobby McChesney
Joseph Bohn
James Kochuba

In the IBM® PureApplication™ System environment, a virtual application pattern is the critical element that allows a user to rapidly set up and manage cloud application infrastructure; a virtual application pattern describes an application, and employs the tools to build the appropriate infrastructure for the application and then deploys the application to that infrastructure. IBM PureApplication System virtual application patterns are the captured essence of years of application-infrastructure-deployment experience and best practices. Virtual application patterns encapsulate optimized solutions of multiple middleware elements to host specific types of application workloads. In this article, the authors introduce you to virtual application patterns and their place in the ecosystem, describe their components and functions, and provide a basic look at how to create and use a virtual application pattern.

The introduction of the IBM® PureSystems™ family takes cloud computing to new heights. IBM PureSystems — in the form of IBM PureApplication System and IBM PureFlex™ System — is an expert, integrated cloud system that encompasses applications, services, hardware, and even the expertise — delivered in the form of best practices patterns — to integrate, deploy, and maintain an enterprise-level cloud environment.

The IBM PureApplication System architecture supports three models of segments of middleware, three different types of workloads used to deliver cloud solutions that are critical underpinnings of the system:

- The use of virtual application patterns through workload platform services. A virtual application represents a collection of application components, behavioral policies, and their links. Using this application-centric definition of the workload the IBM PureApplication System will automatically construct the necessary infrastructure and middleware resources to provision and continually manage this virtual application.
• The use of **virtual system patterns** through virtualized middleware services. Virtual system patterns are a logical representation of a recurring topology for a given set of deployment requirements. For example: A WebSphere® Application Server Cluster pattern containing IBM Deployment Manager, one or more custom nodes, IBM Http Server, and configuration scripts for installing applications to the topology. Using this approach the detailed middleware configuration is explicitly defined and IBM PureApplication System will provision the exact system defined in this virtual system pattern.

• The use of **virtual appliances**, through virtualized infrastructure services. A virtual appliance represents a single server workload instance a virtual machine image file consisting of a preconfigured operating system environment including necessary middleware and application elements pre-installed in an Open Virtualization Format image and application.

The developerWorks community will provide resources to explain these elements. This article describes one of these key components, the virtual application pattern, and how it affects IT professionals.

**The big picture**

Basically, IBM PureApplication System has integrated hardware and software which combines virtualized workloads and scalable infrastructure. Included are middleware support for data and runtime along with and the deployment and management capabilities that streamline and speed these activities, making them more efficient.

Virtual application patterns are constructed with application-centric design. They provide a mechanism to represent middleware applications in a simplified model that abstracts away the underlying middleware infrastructure. These patterns are implemented using virtual application pattern types that integrate the capabilities of multiple middleware software elements into a cohesive, built for purpose solution to allow you to represent your complete, often complex, environments as a single deployable unit.

Virtual application patterns represent a new cloud deployment model and are an evolution of the traditional topology patterns supported in virtual system patterns. Fundamentally, virtual application patterns raise the level of abstraction one notch higher than virtual system (topology) patterns and put the focus on the application. That means, when you use a virtual application pattern, the focus is on the application instead of the application infrastructure.

Virtual application patterns encapsulate the installation, configuration, and integration of middleware, as well as the installation and configuration of applications that run on that middleware. Most of this is completely hidden from you, the user. This means you have less control over configuration and integration, but you also have significantly reduced labor and increased freedom and agility. You can concentrate on the development of the application and its components and let IBM PureApplication System create and manage the infrastructure that services that application.

Reducing deployment time, increasing consistency, and fostering agility are benefits you would likely expect when exploring cloud-based approaches for your middleware application.
environments. The IBM PureApplication System solution tackles these issues making the deployment of cloud middleware environments fast, repeatable, and efficient.

The pattern-based approach is the foundation of IBM PureApplication System; it is consistent for both virtual application patterns and virtual system patterns. Using the cloud appliance, you build and deploy patterns that represent your completely configured application environments. When you are ready to use a particular application environment, you simply pick a pattern and deploy it. IBM PureApplication System automates the deployment, configuration, and integration of the various virtual machines that make up your environment and delivers the completed product in a matter of minutes.

**The elements and functions of a virtual application pattern**

IBM PureApplication System ships with a pre-built set of virtual images, virtual system patterns, and virtual application pattern types that can be immediately deployed. However, creating your own custom patterns from these images and pattern types will further enhance the value you receive from this approach. In that regard, IBM PureApplication System provides comprehensive customization techniques for both pattern models it supports.

The virtual application capabilities in IBM PureApplication System are based on the concept of standardized application-centric pattern solutions. The use of standard patterns allows developers of applications in cloud environments to focus on the application and its requirements instead of the middleware infrastructure and the often-time complex configuration of the middleware products needed to deploy them.

Virtual application patterns define the resources required to support virtual applications, including web applications, databases, user registries, and more. These patterns are the deployment unit for a virtual application.

For example, the web application pattern type provides common application components such as web applications, enterprise applications, databases, and user registries. Along with these common components you can define relationships between components and policies to specify both functional and non-functional requirements for the application. These application components, policies, and links are then interpreted by the IBM PureApplication System in conjunction with functionality provided by the web application pattern type to construct and manage a solution for this application.
Pattern types and plug-ins

The underlying structure for virtual application patterns are pattern types. Pattern types are the containers of solution-specific and topology-specific resources that are required for different types of virtual applications. The pattern types also provide shared services that incorporate runtime services, such as caching service and elastic load balancing. Pattern types are really the aggregation of various capabilities for a specific type of application. The actual solution-specific intelligence is delivered via plug-ins. A plug-in can participate in multiple pattern types; however a plug-in will always have one primary pattern type.

Plug-ins provide all of the necessary functionality to create and manage the real entities that will be realized for the components, links, policies, services, and other features. Plug-ins first provide the visual elements that you see in the virtual application builder when constructing your virtual application pattern. The plug-in is also responsible to provide functions that are necessary to build the model of the system that will eventually be deployed to the cloud. The plug-in provides the necessary scripts to provision and configure the particular application elements. It also includes the logic to federate the necessary elements, surface and react to changes in the configuration, and provide dynamic processing in support of policies.

IBM PureApplication System provides many utilities to make this easy and orchestrates the interaction with the plug-ins to deliver the necessary function in support of the application. This is all optimized and automated within the pattern so that a typical user of the pattern need not understand all of the intricacies of the middleware but can instead focus on the desired behavior of the application.

The default virtual image

A default virtual image is leveraged for all virtual application pattern components. The default virtual image is included in the IBM PureApplication System virtual image catalog alongside other virtual images used with virtual system patterns. However, the default virtual image is special in that it does not include any preinstalled middleware and is a common virtual image, leveraged for all virtual application deployments.
You specify the default virtual image for each hypervisor type in the Default Deploy Settings. The default virtual image can be customized to meet the specific requirements of your enterprise. It provides the functional environment that enables the virtual application patterns to deploy, run, and be managed by the appliance. This environment contains activation code that sets up communication with the appliance during the different stages of the deployment boot process and management of the virtual applications. The image also contain code that is required by the appliance in support of all patterns for shared services, such as load balancing and caching.

**An example, the Web Application pattern type**

One of the initial virtual application pattern types provided in IBM PureApplication System is a Web Application pattern type created for typical online transaction processing web applications. For the purposes of demonstration, the remainder of this article will use this example to discuss the concepts involved in virtual applications and the pattern types that support these virtual applications.

The Web Application pattern type is an IBM PureApplication System extension that you can use to build online web-application-style virtual applications. It provides a set of components that are typical for web applications, such as Java™ Enterprise Edition (Java EE), DB2® provisioning, database JDBC connectivity, Lightweight Directory Access Protocol (LDAP) user registries, and Java messaging.

The Web Application pattern type includes plug-ins for WebSphere Application Server to run web archives files (WAR), enterprise archives files (EAR), and enterprise bundle archive applications and plug-ins (OSGi EBA) that configure connections from applications that are hosted in WebSphere Application Server to existing resources such as databases, web services, MQ, CICS®, IMS™, or LDAP servers. The pattern also includes policies for configuring dynamically scaled server provisioning, load balancing, and caching.

The Web Application pattern type includes several sample application patterns that are ready to deploy so that you can get accustomed to virtual application patterns and their behavior. You can find these samples in the virtual application pattern view. This is the same location where you can begin to create your own custom virtual application patterns from scratch or by cloning the samples provided. Figure 2 shows the virtual application patterns view.
In this view you can see the three sample virtual applications delivered with Web Application pattern type. The selected sample is the Secured Java EE web application and you can see the details of that sample in the right pane. The Daytrader pattern at the top of the list is not a sample but a custom pattern that was created by cloning the Simple Java EE web application and making some minor changes in the virtual application editor.

You can see information about the virtual application pattern in the details panel when selected. To edit the pattern you can select the **Open** icon to enter the virtual application builder and edit the pattern components, links, policies, and properties.

**The virtual application builder**

There are three primary elements that contribute capabilities to a virtual application pattern type — components, links, and policies. Figure 3 shows a screen shot of components, links, and policies in a sample pattern.
Component

A component represents a middleware capability that is needed by the virtual application instance. Typically, a middleware application contains artifacts such as web-based components, database schema definitions, and user registry specifications. When you include these components in your virtual application pattern, it informs the system that instances of these capabilities must be created when the virtual application pattern is deployed.

For example, some of the more common components you’ll find in the Web Application pattern type:

- Enterprise application, such as WebSphere Application Server
- Web application, such as WebSphere Application Server
- Additional archive file
- Existing Web Service Provider Endpoint
- Policy set
- An OSGi application, such as WebSphere Application Server
- External OSGi bundle repository
- Database, such as DB2
- Existing database component
- Remote database component
- Existing IMS database
- User registry, such as a directory server
- Messaging services
- A CICS transaction gateway
- Existing IMS TMRA
- A generic target

Links
Links connect components. Links represent some dependency or interaction between components. For example, certain components in the Web Application pattern type support links to other components in other pattern types, such as to a Database component in the IBM Database Patterns type.

A link serves several purposes:

- It ensures that both the origin and destination components of the link are properly configured to support the connection.
- The link also ensures that the network and firewalls are configured appropriately to allow communication.
- Finally, the link ensures that dependencies are honored and supported so that components can appropriately react to dependencies changes or failures.

Policies
A policy represents how a middleware service is configured during deployment. For example, you can attach an optional Quality of Service (QoS) policy to the virtual application. Two virtual applications might include identical components, but require different policies to achieve different service level agreements.

When policies are added to the application, you can extend the capability of the application. For example, if you want a web application to be highly available, you can add a scaling policy in the virtual application builder and IBM PureApplication System creates the application and topology to achieve that requirement.

Common policies you'll find for the Web Application pattern type:

- Scaling policy
- Routing policy
- Java virtual machine (JVM) policy
- Log policy

Policies can affect the number of virtual machines that are started. For example, if you attach a scaling policy, multiple application server instances are connected together along with a load balancer and optionally, an IBM WebSphere eXtreme Scale server for sharing sessions. Application artifacts are then deployed by starting the components and configuring them appropriately.

Developing a virtual application pattern
Let's say that your organization has quite a few applications in production and more coming down the pipeline. The typical process for IT is to figure out the hardware and software requirements for each application, file requisitions to obtain them, followed by the tedious process of setting up, installing, and configuring the hardware and software.

Next comes the phase when you try to set up monitoring and failover, as well as a way to collect and monitor logs. You may end up writing custom scripts for each application so that things are
easier the next time around and end up building a huge disparate collection of hardware, software, applications, scripts, etc.

Suppose you can standardize the needs of the applications and have enough head room to quickly grow the number of applications, not just in production but also in development and testing. This is exactly where IBM PureApplication System comes into the picture.

What if you had a way to deploy and manage a typical application that has some code deployed in a J2EE application server with some data in a relational database and there’s a quick and easy way to set up logging, monitoring, failover, etc.? With IBM PureApplication System you have just such as solution. Let’s describe a typical Web Application pattern. Artifacts you’d normally find in a Web Application pattern-based virtual application include:

- J2EE Enterprise Archive (EAR) or J2EE Web Application Archive (WAR) for deploying in an application server.
- Script for creating database schema/table/rows for initializing a database.
- List of users and groups as an LDIF file (LDAP Data Interchange Format).

As we’ve already discussed the web application pattern type already includes all of these components in an easy to consume, optimized, and fully integrated fashion. You can very simply build your pattern using the virtual application builder with a simple drag-and-drop user interface. With a system virtual application pattern like this, you can quickly design a virtual application, upload these artifacts, specify policies with respect to logging, monitoring, scaling, and deploy the virtual application in your private cloud.

IBM PureApplication System contains an easy way to build and edit a virtual application pattern: The Virtual Application Builder. It is one of the elements of a virtual application described in Figure 3.

The editor is very easy to use. It works like this: You create a new virtual application, drag and drop the Enterprise Application component and a Database component and link them together on the Virtual Application Builder canvas. You upload a JEE-based EAR or WAR that contains your code artifacts and a database schema that describes the table structure expected by the code. Because this pattern uses components from components in the Web Application and IBM Database Patterns, IBM PureApplication System will leverage the knowledge included in those components using the Web Application and IBM Database Patterns in an optimized fashion.

One of the main distinctions of virtual applications as compared with virtual systems is that you do not decide or need to know exactly how many virtual machines end up getting started on the hypervisors to construct the running system. You can delegate this responsibility to IBM PureApplication System which looks at the components, the links between them, and the associated policies to figure out the best fit in terms of number of virtual machines and what exactly runs on which virtual machine.
With a virtual application you simply specify policies to govern say the scalability aspect of your application when they design the application. You don't need to pick the middleware component that is needed and how many of each.

Once deployed your application is automatically managed for the high availability and will honor the policies that you have specified for the running state. You view into the application is provided via a fully integrated, optimized user interface. There is no more switching back and forth between different administrative consoles to manage and configure the system.

**Using a virtual application pattern**

You might be asking yourself "what conditions suit the use of an IBM PureApplication System virtual application pattern over a virtual system pattern?" In this case, let's further define the question so that it makes more sense.

In essence, virtual application patterns are similar to virtual systems patterns in that they both represent models of a virtualized application for deployment to the cloud. When we refer to patterns in the context of cloud, we are referring to the encapsulation of installation, configuration, and integration activities that make deploying and managing environments in a cloud much easier. Regardless of what kind of pattern you end up using, you benefit from treating a potentially complex middleware infrastructure environment or middleware application as a single atomic unit throughout its life cycle of creation, deployment, and management.

Let's consider the cloud tradeoff continuum in order to understand the differences between the application and the systems pattern deployments. (Again, we've used IWD since it is the immediate predecessor technology and we have lots of data and experience using it.)

**Figure 4. The cloud tradeoff continuum**

- The X-axis represents the degree to which you have customization control over the resulting environment. The degree of control gets lower as you move from left to right.
- The left Y-axis represents total cost of ownership (TCO) which decreases as you move up the axis.
• The right Y-axis represents time to value which similarly decreases as you go up the axis.

Naturally, enterprises want to move up the Y-axis, but they are sometimes hesitant to relinquish much control (move to the right on the X-axis) in order to do so.

With what this figure shows as a reference point, start to think about the two patterns-based approaches a bit more.

**Scenario: Using a virtual system pattern**

Consider a fairly simple web service application that you want to deploy to the cloud. If you were to use a virtual system pattern to achieve this, you would probably start by using parts from the WebSphere Application Server Hypervisor Edition image to layout the topology. You may include a deployment manager, two custom nodes, and a web server.

After establishing the topology, you would add custom script packages to install the web service application and then configure any resources the application depended on. Users that wanted to deploy the virtual system pattern would access it, provide configuration details such as the WebSphere Server cell name, node names, virtual resource allocation, and custom script parameters, and then deploy.

Once deployed, users could access the environment and middleware infrastructure as they always have. That means they could run administrative scripts, access the administrative console provided by the deployed middleware software, and any other configuration action they would normally want to perform.

**Same scenario: Using a virtual application pattern**

Using a virtual application pattern to support the same web service application would result in a markedly different experience from both a deployment and management standpoint.

In using the virtual application pattern approach, a user would start by selecting a suitable virtual application pattern type based on the type of application type. This may be a pattern type shipped by IBM, such as the IBM Workload Deployer Pattern for Web Applications, or it may be one created by the user through the extensibility mechanisms built into the appliance.

After selecting the appropriate pattern, a user would supply the web service application, define functional and non-functional requirements for the application via policies, and then deploy.

The virtual application pattern and IBM PureApplication System provide the knowledge necessary to install, configure, and integrate the middleware infrastructure and the application itself. Once deployed, a user manages the resultant application environment through a radically simplified lens provided by PureApplication System. It provides monitoring and ongoing management of the environment in a context appropriate for the application.

This means that there are typically no administrative consoles and users can only alter well-defined facets of the environment. It is a substantial shift in the mindset of deploying and managing middleware applications.
So how do I decide?

The main point is, when you deploy software using a virtual system pattern, for the most part you manage the environment the same way you have always managed it for that particular type of software; normally via administration consoles. With virtual system patterns, you don't focus on changing the way you operate or manage the software; instead, you primarily focus on improving the delivery of said software.

With virtual application patterns, you fundamentally change everything about those environments. You are working with a highly optimized and automated solution. The burden of managing high availability and dynamically reacting to changing conditions is built into the pattern type solution so you only need to specify your business level requirements. The management and operations of that environment are completely integrated in the IBM PureApplication System user interface. Everything is integrated and highly specialized for the specific application type.

One important thing to note is that IBM PureSystems supports both of these models concurrently and can actively manage both types of patterns. You should base your decision on your requirements and management needs for the particular application that is being considered.

For each application, you need to decide whether you favor the middleware-infrastructure-centric approach of virtual system patterns or if you prefer the application-centric approach of virtual application patterns. You may be driven by the need to support very specific configurations that don't fit easily into an already available virtual application pattern type. In that case you may choose to create your own virtual application pattern type or use a virtual system pattern to create the exact topology that your application requires, perhaps even replicating a physical environment that you previously implemented. In other cases you may find that your application fits well into one of the virtual application pattern types that are already provided.

Whenever possible you should seek to leverage the optimization and convenience of a virtual application pattern as this will always provide the lowest total cost of ownership and shortest time to value. However, there will certainly be scenarios where you require very detailed configurations and therefore decide in favor of the detailed control that is available with virtual system patterns.

The most important thing is to understand all of the options and make an informed decision. Look at your use case, understand what's available to help you accomplish that use case, and finally, decide on what you want your user's experience to be.

Finally, it is also important to note that IBM PureApplication System supports both of these models concurrently. You can have a mix of virtual applications, virtual system, and even virtual appliances all deployed to the same pool of cloud resources. The robust capabilities built into IBM PureApplication System enable this variety of deployments, allowing you to choose the deployment model for each application to deliver the best fit with this highest return on investment.

In conclusion

We've introduced you to virtual application patterns and their place in the IBM PureApplication System ecosystem (as well as how they compare and contrast with virtual systems patterns and
virtual appliances); we've also described virtual application pattern components and functions, and provided a basic look at how to create and use a virtual application pattern.

For more experience in pattern-centric cloud deployment, take a look at the companion communities, forums, and articles in the Resources section.