EXPERT ADVICE

The Fast Track From Code to Cloud

By Nigel DeFreitas
TechNewsWorld
08/25/10 5:00 AM PT

Back to Online Version
E-Mail Article
Reprints

The evolution of software libraries, development environments, and dynamic language platforms now provides us with the tools to develop applications faster, with better unit and continuous testing support. As cloud computing solutions and application lifecycle management tools evolve, agile processes will adapt to accommodate the speed at which new stories can be continuously delivered.

Over the past decade, software development has undergone several transformations that have led to the new practices and methodologies such as agile development. Combining emerging technologies -- including dynamic language platforms, cloud computing, and agile development practices -- offers a paradigm shift away from traditional software development methods.

Current industry trends related to agile development are likely to evolve in the next several years to accelerate the time to market for software development projects.

Libraries and Application Servers

Allan Kay, a famous computer scientist, once commented, "The best way to predict the future is to invent it." Developers in the late 1990s had at their disposal a few primary programming languages for building Internet applications. The majority of the applications were client-server based. Most of them used software-based SQL engines, XML parsers, and Java runtimes.

There were relatively few open source projects and components in widespread use; however, over the years, several specialized ActiveX and Java code snippets became available that solved

specific low-level problems. By 2005, entire frameworks emerged that aggregated several libraries into a more comprehensive set of tools. These frameworks provided consistent resource management and solved common problems. It was no longer necessary to write your own low-level plumbing code.

Today, many open source libraries and frameworks are being assembled into entire platforms, such as the OpenStack (open source Platform as a Service, or PaaS) and OpenNeblula (open source Infrastructure as a Service, or IaaS) projects. In addition, hardware appliances have since emerged to subsume common functions such as SQL engines, DOM parsers, and JVM runtimes.

These will become commonplace in the future and allow developers to perform operations at near wire speed on terabytes of data. More than ever before, this evolution of libraries has freed up developers to focus more on business requirements and framework/library selection, rather than writing low-level plumbing code or integrating code snippets.

In the future, PaaS solutions will evolve to provide all the amenities now available in application servers and frameworks, such as messaging and transaction management. As applications mature, developers will spend less time contemplating scalability and other low-level issues.

Future applications will be built much faster and will easily process significantly larger volumes of data. Organizations that do not adopt PaaS and IaaS solutions in favor of traditional alternatives won't be able to compete. IT executives will implement solutions that save the corporation money and provide better agility.

Improvements in Web Application Productivity

Early Web application development techniques involved authoring HTML and ASP, JSP or PERL scripts. Developers' primary tools were text editors and IDEs that are no longer in mainstream use today. Tools got better, and soon developers could drag and drop UI elements and database resources to a canvas without writing significant amounts of code.

Today, it's possible to use Eclipse and dynamic languages such as Groovy with the GRAILS platform to build simple dynamic Web applications with as few as seven lines of code. A developer simply defines scaffolding and domain model. All other elements of the UI and associated MVC components are automatically generated.

Spring-Roo, another scriptable tool that brings similar productivity gains to traditional J2EE development, and ASP.NET frameworks such as DotNetNuke offer developers a rich ecosystem of free and commercial extensions for creating composite applications -- without writing a lot of code.

By using built-in continuous testing capabilities, developers get feedback as soon as a feature breaks. It's even possible to deploy an application directly to a cloud platform from right within development environments. As a result, developers are now able to focus more on business requirements and innovation, while building complex applications faster and with less effort.

In the future, application lifecycle management tools will provide auto-deploy features that serve

up the latest change sets to cloud-based staging environments. Dashboards will link the change sets back to user stories in an issue tracker and automatically roll out new stories as project owners accept delivered functionality.

These continuous delivery capabilities will result in significantly shorter time frames for delivering user stories -- five-day dashes instead of two-week sprints. Agile projects will become even more fluid. If developers miss just two days of participation, they will be out of the loop. Communication between project team members will be paramount.

Data Growth and Context-aware Architectures

Early Web applications consisted of mostly static content primarily contributed by webmasters. As applications evolved, more content was added by exposing corporate databases. With today's wiki and CMS systems, even more content is contributed by crowds of authors. In 2009, the federal government launched data.gov, which provides access to a variety of raw geodata to the public at no charge.

It's difficult for companies to ignore the exponential growth of information available today -- and all that will be available in the next 10 years. With this new data, companies will find ways to differentiate their products and services from their competitors'. As a result, developers will need to build applications that operate on more and significantly larger data sets.

Specifications such as Service Component Architecture (SCA) will become important for developers when designing atomic reusable components. Some of the components will be important parts of Data as a Service (DaaS) instances. However, not all data services will be created equal. Data streams integral to core company strategy will be given higher priority in every respect.

Applications like Flipboard for the iPad -- which creates personalized social magazines based on content your Facebook friends and Twitter followers find interesting -- are a hint of things to come. With similar platforms for presenting content, it's possible to perform analytics to find out what users are interested in most, and create prominence algorithms for generating and displaying more desirable content.

Companies will seek out similar types of contextual data about their customers that increases the stickiness factor of their products and services. Software architectures will need to become more context-aware. Developers will need to deal with geodata as well as a variety of other contextual information when presenting content in next-generation applications. Software architectures that instill strong SCA and DaaS standards will leverage assets with ease and agility.

The Road Ahead

The evolution of software libraries, development environments, and dynamic language platforms now provides us with the tools to develop applications faster, with better unit and continuous testing support 🛒.

As cloud computing solutions and application lifecycle management tools evolve, agile processes

will adapt to accommodate the speed at which new stories can be continuously delivered.

In addition, the volume of data that applications process in the years to come will increase significantly. It will be critical for new software architectures to have first-class support of features that aid in understanding customer context.

The future is ours to create. What will your invention be? ECT

Nigel DeFreitas is a managing architect at ISO. He is currently investigating entity and relationship resolution and predictive model deployment -- technologies that help identify fraud and bring predictive models to market more quickly.

Next Article in Developers

The IT Project-Tracking Database: Keep Your Key People in the Loop
August 09, 2010

A project-tracking database should not just be a nominal entity that exists for the sake of existence; rather, it should contain some key attributes that every person in the company can draw important conclusions from -- and use them to form the basis of key decisions that impact not just the project itself, but also the overall vision and direction of the company.
