HP World 2001
August 2001

Conference Track:
Mobile and Wireless Computing
Target Audience:
All
Technical Level:
Intermediate

Provisioning Web Services for Mobile Applications
by Bill Ho, Chief Technology Officer, vVault

Introduction

Referring to fundamental changes in the IT market, Hewlett-Packard CEO Carly Fiorina recently observed that increasingly, IT would be "provisioned, delivered, metered and managed and purchased as a service."[1] Indeed, a new framework for deploying IT solutions is essential.

For both traditional service providers and the growing class of hardware providers joining their ranks, this new era of accelerated application development cycles and the proliferation of new mobile devices and standards demands flexible, distributed, and rapidly integrated solutions in order to compete and win. For IT managers responsible for deploying corporate solutions, applications must also be easy to manage, upgrade, and maintain. In this paper, we look at the current environment for mobile computing and the challenges of provisioning it to your workforce efficiently and reliably.

Web Services Model

As a group, device manufacturers and wireless network services face a challenging and competitive environment. These companies are hungry to provide useful applications and capabilities that drive usage and adoption, while their application service provider counterparts show hesitance in supporting new devices until they observe significant usage, adoption, and market share – a classic chicken and egg problem. One way to alleviate risks on both sides is to adopt a Web Services model of application provisioning. For the client, the distributed nature of the Web Services model makes integration easier and more dynamic. Service providers can publish a standardized interface, keeping the

[1] "HP warns IT spending slowdown is now global", Paul Taylor, the451.com, Jun 06, 2001.

coupling between applications loose rather than tightly bound, and support multiple clients with little or no customization.

Simply stated, Web Services is a framework for applications to communicate and work with other applications over the Internet. Sophisticated and complex applications can be built and managed by leveraging services distributed throughout the Web.

Although the Web Services model works well in both the wired and wireless world, it is especially relevant to the mobile environment. The sheer number of devices, networks, protocols, and operating systems in existence and emerging daily generates a complex web of development trees. These new interfaces and devices pose a continuing challenge for developers of mobile applications and the service providers deploying them. Additionally, for enterprises, the delivery of applications via Web Services has the potential to simplify provisioning and open up more flexible payment models – including pay-per-use.[2]

vVault's own mobile and Web delivery platform for business services powers device and network agnostic end user applications that integrate file access and management with messaging services such as email and fax. From inception we employed a design methodology that considers both the needs of service providers who will distribute our technology, and the end users who will benefit from it. By adopting a Web-services platform approach, vVault is able to address our customers' needs around customization and integration quickly and flexibly. In addition, we can support them in the future as their business needs and end-user offerings evolve.

This paper presents some of the principles vVault has found useful in building a modular and extensible services platform within an evolving technology and business environment.

Mobile Application Design Considerations

The delivery of mobile applications introduces an assortment of issues. At this early stage in the evolution of mobile technology, a foremost requirement is flexibility.

- Device/Network/Operating System Independence - At a minimum, flexibility refers to the ability to provide easy upgrades to existing back-end technologies in order to support new devices, networks, and communications systems as they come online.

[2] "CTO Forum: Apps on tap model to thrive despite challenges", Cathleen Moore, Infoworld.com, June 21, 2001.

- Customer Requirements - Flexibility also refers to support for customers' unique needs for custom solutions and integration with legacy systems. In addition, applications must be re-designed for ease of use in a size or capability-constrained environment (particularly with smaller devices) with minimal overhead and maximum efficiency.

- Extensibility - Finally, flexibility refers to the ability of the application developer, partner, or third party to extend the their application in order to introduce new features and capabilities easily.

vVault uses three mechanisms to ensure flexibility of our platform, today and beyond (see Table 1):

- Open standards to ensure future compatibility and to facilitate communication with diverse technologies;

- An Integration Framework (Application Programming Interface) to standardize access to functionality; and

- A modular platform architecture to allow for highly customizable solutions and drop-in extensions of the platform's capabilities.

Table 1: Design Principles to Ensure Flexibility

Mechanism	Client Benefit	Service Provider Benefit
Open Standards	Well understood and defined protocols and transport mechanisms	Facilitate communication with heterogeneous clients and applications
Integration Framework	Underlying technology is abstracted, access methods are simplified	Easy to maintain and upgrade with minimal impact on existing clients
Modular Platform Architecture	Ability to pick and choose modules to build customized solutions	Ability to drop in new modules to extend the platform's capabilities

Open Standards

vVault uses several emerging standards to define the communications layer for our services, including many of the implementations based on the work around the W3C's XML Protocol. The XML Protocol establishes communication between

two peers over a distributed network (i.e. the Internet). vVault's implementation utilizes SOAP as a request-response mechanism, XML as a message delivery format, and WSDL to describe the available services and input and output parameters. The core services that make up the middle tier of vVault's platform are written as Java beans. We selected these technologies in order to support an extensible yet simple messaging format that is client agnostic and has support for any language, platform, or operating system that can construct and receive valid SOAP messages.

XML is useful on the front end as well – it standardizes data delivery and helps support multiple clients. Each client can parse the format-neutral raw data and render it in the appropriate format, whether it's for a Web browser, cell phone, or PDA. As new devices or front end clients are introduced, only lightweight rendering engines need to be built, and minimal or no back-end changes are required.

There are disadvantages of using leading edge technologies. They tend to be immature and consequently require that developers compensate for these inadequacies. One such example, which became evident during the development of the vVault API, was the lack of support for non-primitive data types. WSDL, at that time, had no mechanism for dealing with complex data types such as arrays. Consequently, once the code generation process had been completed, we spent a significant amount of work to integrate support for the array data types.

Integration Framework

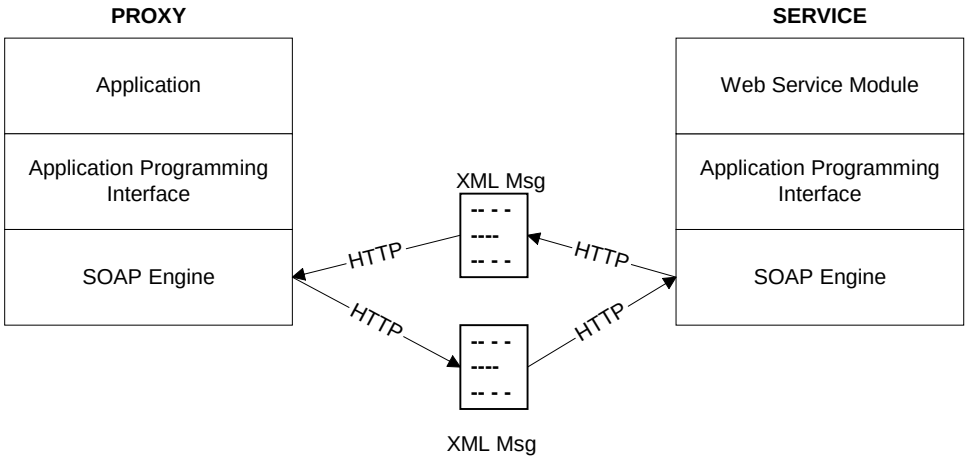
To support custom integration today and in the future, vVault exposes its data services and capabilities through a robust API. The delivery of services via an API has several advantages - it is:

- flexible (client- and network-agnostic interface)
- lightweight (making for easy integration)
- modular (allowing custom solutions and incremental deployment)
- extensible (maintains backward compatibility with existing users while supporting new users and applications)

As noted above, vVault uses the Web Services Definition Language (WSDL) to describe available services represented by our API. Based on WSDL definitions, we generate code stubs that specify the transport mechanism (e.g. HTTP, SMTP, etc.), and create the appropriately formatted XML messages (header and body) for service requests and responses for each method we expose through our API.

From the application's perspective, all requests look the same and consequently can be handled in a consistent manner (see Figure 1). Additionally, back-end improvements are transparent to the client; clients can leverage new services more quickly as we deploy them because we maintain a consistent method calling format. We call this being "backward compatible" but "future friendly".

Figure 1: Use of Open Application Programming Interface



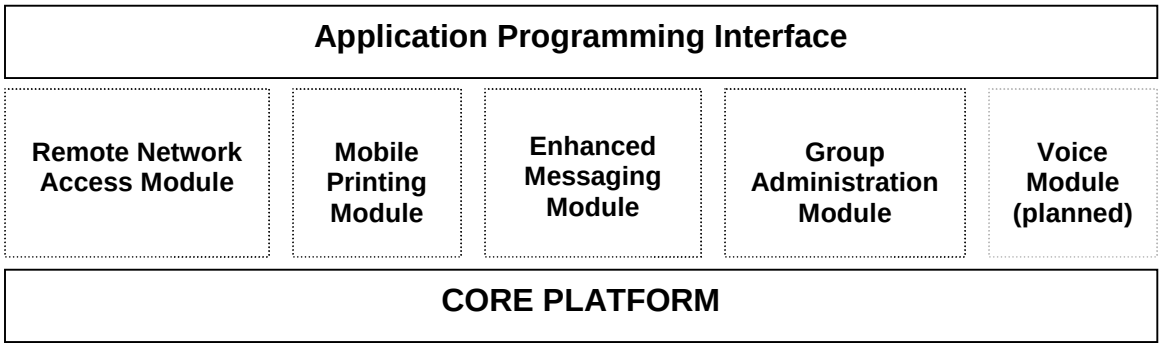
Modular Architecture

Web Services is not a static model – services can be scaled, upgraded, replaced, and extended. Web Service providers can achieve scalability, not only in a technical sense, but also from a business perspective by adopting a modular platform architecture.

For example, the vVault platform utilizes a "Core" platform – or engine – that powers functional modules (see Figure 2). vVault's core platform provides for redundancy, security, and reliability, as well as a robust communications infrastructure.

At the applications level, vVault separates the platform services into major functional groups (modules) and standardizes the interface for communication, parameter passing, and data sharing between modules. Thus, the various functional modules are tightly integrated, yet independent and fungible. This internal API allows us to maintain modules, upgrade to new technologies, and add additional services quickly, without affecting existing modules.

Figure 2: Example of vVault's Modular Platform Architecture



vVault's modular architecture enables application developers to design a basic custom solution and add new modules of functionality incrementally thereafter. Thus, as an application grows and needs new functionality, new services can be added quickly, both through extensions of vVault's platform, and through other Web Service providers' platforms as well.

Conclusion

The development community and several major technology companies are increasingly embracing Web Services as a means of harnessing external functionality to build new applications, extending the functionality of current applications, and providing rapid support for new client and device intefaces.

Current efforts in Web Services deployments are nascent, but the strong support of Web Services is sure to bring about vast improvements in the development tools used to build and deploy them in the near future. Today's tools are functional and help alleviate many of the more tedious aspects of the lower level communication between client and server. Next generation tools promise to add more complex data types, generate better code, and support additional programming languages. For businesses, potential benefits range from ease of provisioning of IT services to more flexible payment options.

With continued advances in the underlying technologies such as WSDL and SOAP – used in conjunction with a Web Services delivery and management architecture – we believe that a new generation of mobile applications providers, managers, and end-users will benefit significantly by moving to this powerful and elegant development and service delivery model.

Bill Ho is the founder and CTO of vVault. Founded in June 1999, vVault is a privately held, San Francisco, CA based developer of the award winning vVault Platform, a leading infrastructure technology for carriers and other communications companies seeking to deploy mobile productivity applications.