The Webservices Architecture for Mission Critical Systems: A Case Study

John Peppiatt

Manukau Institute of Technology
Manukau City, New Zealand

John.Peppiatt@manukau.ac.nz

Proceedings of the 15th Annual NACCQ, Hamilton New Zealand July, 2002 www.naccq.ac.nz

ABSTRACT

At the 2001 NACCQ conference we presented a paper on design patterns for CGI web applications with Visual Basic that explored methods of using VB and Access to create fast, robust web applications with low-cost hardware and software (Lopez and Peppiatt, 2001). We have now implemented a model derived from this to develop and deploy a widely distributed bookings and attendance tracking system. This was an opportunity to apply the techniques previously explored in an experimental situation to a real, high profile project.

The project required the provision of a rich GUI to users in sites throughout New Zealand manipulating data held on a central server at MIT. The system had to be developed, tested and deployed in under 2 months - with the usual limited budget. At the time of development commercial Web-Services architecture products like Delphi 6 and .NET were only available as Beta. We developed a custom marshalling protocol over HTTP, and methodologies to manage disconnected record sets and synchronisation.

This paper describes the technology used and the implications for systems development in the emerging web-client/web-service world.

1. INTRODUCTION

At the NACCQ Conference of 2001, we presented a paper exploring the capabilities of VB, CGI and COM as technologies for creating robust, high performance web sites (reference). A part of this research involved the simulation of the use of a web site by generating messages with Microsoft's Internet Transfer Control. In August of that year an opportunity presented itself to develop a highly distributed message processing system for managing computer room bookings for a nationwide community education scheme (Key4Free). This paper discusses how this was achieved and the implications for developing other systems using, what is now commonly called, WebService technology.

2. SYSTEM REQUIREMENTS

The main objective was to provide a rich GUI to end-users to manage client registrations and bookings stored in a central database. A number of sites would be built and commissioned over one to two years with a target of around 25 locations throughout New Zealand. Each remote location would have a single user. In addition, a central call center with around 5 call center operators had to be supported. The system has to cater for tens of thousands of clients and with up to two hundred thousand bookings on file at any

one time. Response time had to be 'as fast as possible'. The system had to be designed, developed and implemented within two months and with minimum capital expense.

3. SYSTEM ARCHITECTURE

Our previous research on performance of IIS, COM and the Internet Transfer Control using Access as a database gave theoretical support for this using this technology mix. At the time of the development Microsoft's SOAP based technology (WebServices) for developing such systems was only in beta, the other commercial candidates known to the author were Delphi 6 and JBuilder each with some WebService support. The WebService technology however only simplifies the marshalling of remote procedures and data transfer; it does not address the real challenges of distributed design. The decision therefore was to proceed with some simple in-house marshalling technology and develop a client server system using VB generated COM components.

The schematic shows the client and server side components (see Figure 1).

The client side COM object is packaged as an OCX to allow hosting either in a Web page or VB Form container (in production a form is used). The server side is a single in-process COM object called through an IIS active server page for each message. The only special feature is the use of HTTP headers to marshal some requests - this did require some re-configuration of the Linux Squid Internet proxies used in remote locations.

4. THE MAIN CHALLENGES

To provide a rich and responsive GUI in a highly distributed environment the main issue is keeping enough information client side to minimize the number of times the server must be contacted. Two strategies were used to achieve this:-

- A compact and denormalised form of booking information is cached in memory on the client side. Time stamp techniques are used to keep this cache reasonably current
- Business constraints are checked initially on the client side and then again on the server side to give more immediate feedback to the user where possible.

5. IN PRACTICE

Development time was around 100 man-hours. The architecture has proved itself exceptionally robust running without interruption for many months. Data volumes have reached 9,000 clients and over 50,000 booking records. A slight slowdown on the client side did occur in the cache management when large numbers of changes were being merged into memory held record sets. This had to be optimized. Processing time at the server is remarkably low; a single corporate standard workstation is not stressed. In practice a second workstation is connected and on hand for backup purposes.

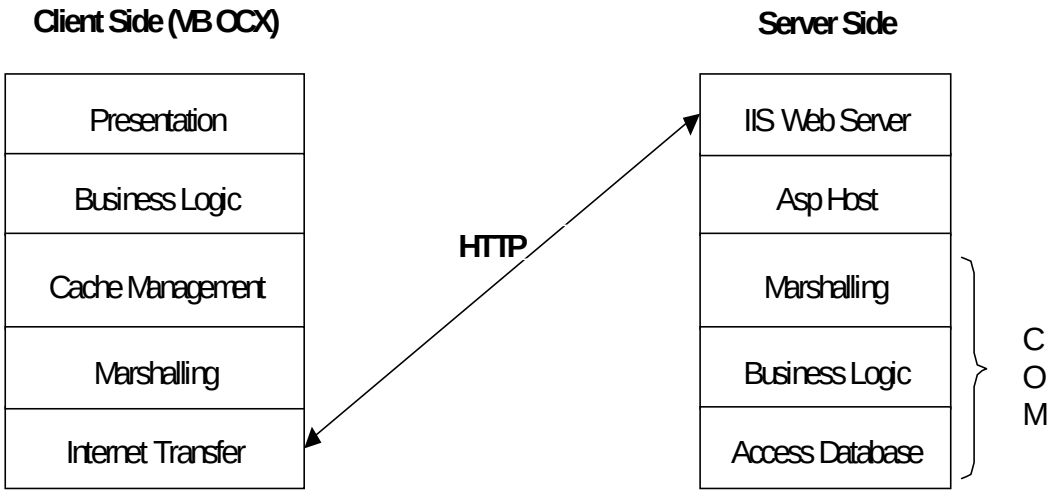

Figure I

6. THE WORLD OF WEB SERVICES

The emerging requirement for highly distributed systems based on this WebService model poses problems for programmers and analysts alike.

- The analyst must be increasingly aware of the characteristics of these platforms and accommodate their behaviour into use case design. In particular, the need to design system which work on disconnected datasets which perform business rule checks twice and not once adds a layer of complexity to use case modeling especially with the number of alternate paths that must be catered for.
- The programmer must understand the performance characteristics of remote procedure calls executing either synchronously or asynchronously with respect to the main process flow. Cache management is only partly standardized and will continue to require application specific techniques, which in turn frequently require a move from normalised data to denormalised forms.

As with most emerging technologies, practitioners will develop design patterns they trust to serve a problem domain and this accumulated wisdom will then feedback into more formalised and packaged techniques for the next generations. Given the mixed skill set in the professional world the industry may be in for a bumpy ride!

7. CONCLUSION

The author's first experience into highly distributed systems was a very positive one but only made so by careful modeling of the technical aspects beforehand. The experience highlights new challenges for system analysts and technical designers; the author anticipates that many will over-look the important issues in the mad rush to develop distributed systems. The WebService tools now available in many RAD tools do simplify marshalling of data and provide the client to server side link in the development process. For many problems however, these RAD tools do not yet provide complete design frameworks to be used for the majority of situations.

REFERENCES

Lopez, M. and J. Peppiatt. (2001). Design Patterns for Cgi Web Applications with Visual Basic. in 14th Annual Conference of the NACCQ, Napier.