SPCQuote: From Smart Client to Distributed Processing

Karen Hope
The St. Paul Companies
5801 Smith Ave.
Baltimore, MD 21209 USA
011.410.205.1194
Karen.Hope@stpaul.com

John Finegan
The St. Paul Companies
5801 Smith Ave.
Baltimore, MD 21209 USA
011.410.205.5857
John.Finegan@stpaul.com

ABSTRACT

This paper describes the evolution of St. Paul Fire & Marine's commercial insurance policy writing system from a smart-client Smalltalk application to an n-tier distributed service model supported by two Smalltalk dialects, Java servlets running within Websphere, and XML as a means of data abstraction.

1. Background

The Business Foundation System (BFS), deployed in 1996, was designed to support the complete workflow of commercial insurance policy writing for United States Fidelity & Guarantee (USF&G, now part of The St. Paul). It began as a smart-client model based on a VisualSmalltalk Enterprise (VSE) client application communicating to a Sybase System 10 database and initially supported 120 employees in 3 locations. The VSE client application was designed based on object-oriented principles that yielded well-encapsulated subsystems. Behaviors relating to business objects (i.e., aPolicy, aCoverage, aVehicle), user interface, persistence, printing, statistical reporting, and policy rating, though residing in the same Smalltalk image, were conceptually independent of one another. The ability to alter one subsystem without impacting another has greatly facilitated changes within BFS, and since its deployment, the system has undergone over 30 subsequent releases to increase and improve functionality. The most revolutionary of these was SPCQuote. Released in April, 2000, SPCQuote significantly changed both the system architecture and its development environment while satisfying disparate business objectives concerning application distribution, scalability, and maintainability.

2. Challenges

The initial BFS architecture very successfully met its business objectives and its object-principled design proved efficient and dynamic through subsequent re-architectures and product enhancements. Nevertheless, by 1998, BFS was facing many challenges. Notably:

- The success of BFS encouraged business to increase the number of insurance products offered through the application, expanding the size of the application executable. This growth inconvenienced our users by requiring them to purchase and install more memory for their client machines.
- Our pool of users was changing. No longer supporting just internal employees at three locations, BFS' policy quoting capability was provided to external insurance agents at many locations via either CD-ROM or a Citrix/Internet interface. Neither approach was economical. New CD-ROMs needed to be created and distributed to more than 1,000 agents whenever the application or reference data changed, and each Citrix server was expensive to maintain and could only support 8 concurrent users.
- ParcPlace-Digitalk, supplier of our VSE-Smalltalk development tool, announced it was discontinuing further development of and phasing out support for said tool.
- When USF&G merged with The St. Paul, the combined company found itself with redundant systems and set out to cut costs. BFS needed to prove its value with regard to scalability, adaptability, and cost-effectiveness.

3. Solution - SPCQuote

SPCQuote originated from an effort to address the concern of discontinued product support by porting from VSE to IBM's VisualAge Smalltalk (VAST). With the aid of Synchrony Systems and their SMT product, we performed an internal proof-of-concept that successfully migrated one component of our application from VSE to VAST. Pleased by the quality of the migration but concerned by indications that porting the entire application, an effort providing no additional system capabilities, would require a daunting 12,000 person-hours, we began considering other business needs that could sensibly be combined with a port. This culminated in a very ambitious re-architecture plan that would:

- Port most but not all of the VSE code To VA. The greatest encumbrance in porting the entire application from VSE to VAST was mapping custom VSE UI widgets to VAST graphical components; porting all other components of the application was comparatively simple.
- Replace both the CD-ROM and Citrix BFS interfaces used by external insurance agents with an affordably scalable web-based interface.
- Continue efforts to reduce BFS' client footprint size by splitting the client/server application into separate services.
- Take a bold step in anticipating the future of systems at The St. Paul by designing and moving to production the company's first component-based application.

4. Implementation Features

- **Reuse:** Two system components, one providing domain model update processing (Domain) and another providing business rules evaluation (Edits) and policy premium generation (Rating), were ported from VSE to VAST in order to operate in an IBM Server Smalltalk Environment. This allows (a) the existing UI and VSE domain to support full policy lifecycle processing for internal users, (b) the ported domain behavior running in a Server Smalltalk Environment to support a new browser interface for external on-line quoting, and (c) the ported Edit and Rating behavior running in a Server Smalltalk environment to support both the browser domain as well as the VSE domain. This meant not only that the VSE application could be stripped of the edit and rating components, but also that, by keeping the domain logic supporting the browser interface in Smalltalk (ported from VSE to VAST), we avoided re-implementing five years' accumulated behavior.
- **Java Servlets for Web-based Client Interface:** Servlets running within IBM's Websphere provide extensive framework processing. An HTML-based browser interface should allow us to be able to switch to a graphically richer toolset if our requirements eventually so demand.
- **Distributed Object Communication:** The client application was spread across multiple configurable tiers. Edits and rating, once isolated, provide a reusable cornerstone for disparate input sources. A black-box framework uses RMI (Remote Method Invocation) to enable inter-object communication between Java and Smalltalk. Other communications strategies, such as CORBA, could alternatively be implemented in this layer without affecting client or service behavior.
- **Separation of data and representation:** Back-end services were modified to interpret XML as their data source, ensuring input neutrality to enable the services for requests from the browser as well as from the original VSE client application.

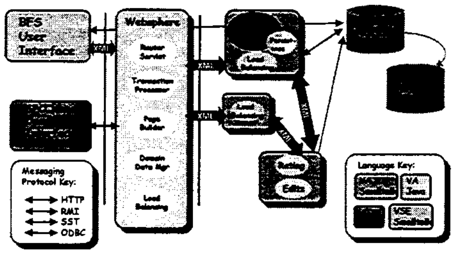

Figure 1: Current BFS/SPCQuote architecture

5. Direct Benefits

Business benefits to Commercial Lines policy writing due to deployment of SPCQuote in April, 2000 were immediate. Over 7,500 agents now have direct access to real-time quoting capabilities with The St. Paul companies. Agency quotes originating from web-based SPCQuote now account for 70% of total quotes booked and billed. Quotes originating from CITRIX have been reduced from 40% to 5%, positioning the servers for retirement. Software distribution tasks have also been reduced, saving $400,000 annually.

6. Recommendations

SPCQuote challenged the organization. Learning curves for new software components as well as development of deployment techniques for the new component-based architecture were embraced simultaneously. The project, including training, was completed in only 15 months. Any organization attempting such an ambitious undertaking could benefit from the following recommendations:

- **Re-architect to the appropriate tool.** For SPCQuote, despite the obvious attraction of porting Smalltalk-to-Smalltalk, the operational limitations imposed by Server Smalltalk greatly diminished the potential benefits of a distributed architecture. As a consequence, we are now faced with porting these same services to a more scalable environment, that is, Java.
- **Expect to be unprepared.** Realize that with a distributed architecture, end-to-end performance metrics are not available until every component is successfully integrated. This limits the possibility for major design revisions during the initial implementation iteration.
- **Ready the organization for the architecture.** In addition to design and user documentation, documentation of implementation, deployment, and debugging techniques should be recognized as deliverables critical to the continued evolution of the application. The technical infrastructure, including production support, and system visibility and control measures must be ready at initial deployment time. Accept that future project planning must account for the increased complexity of testing and integration.
- **Sell the new system to the users.** We found ourselves in the unenviable position of shouldering the costs of cutting-edge technology to replace an existing entrenched application. It was not easy for our users to appreciate our strategic vision when their familiar application was replaced with a less predictable, server-based entity

7. The Future of BFS

Originally conceived as a line-for-line dialect translation yet executed as total system rejuvenation, SPCQuote represents The St. Paul's most significant advancements in systems architecture. Changes introduced by SPCQuote provide us a viable foundation upon which to build the next generation e-commerce application. Currently, we are planning to replace Smalltalk altogether with a more progressive development language and to provide full-policy lifecycle support over the Internet. An effort is also underway to provide BFS quoting through third party interfaces using industry standard ACORD XML. Further, a web-based end-consumer model has been piloted and will be in full production by EOY, 2001.