Collaborative SVG as a Web service

By Xiaohong Qiu, Bryan Carpenter and Geoffrey C. Fox

Introduction

We present a reformulation of SVG and in particular the Batik browser in a Web service structure exploiting the MVC (Model View Controller) paradigm. This work has two major goals; firstly it allows us to make a very flexible collaborative SVG browser exploiting our general approach for collaborative Web Services (http://grids.ucs.indiana.edu/ptliupages/publications/foxwmc03keynote.pdf). Secondly it shows how applications and in particular W3C DOM based applications can be built as Web Services in a case (the Batik SVG Browser) with a well written open source code space. Building applications as Web Services has several important features; it allows automatic generation of collaborative applications; it unifies distributed and local services; finally it allows easier customization of the user interface to support the needs of clients or users. We discussed the latter point in SVG OPEN 2002 (http://grids.ucs.indiana.edu/ptliupages/projects/carousel/papers/draft.pdf), which described PDA interfaces to SVG. In this paper we focus on the Web Service SVG architecture and the structure of the associated event model. We believe that our analysis could suggest architecture of future office and visualization systems, which are built around object models like the W3C DOM.

We have also introduced two useful technology components to support this program.

1) A Web Service that supports collaboration by providing the Web service equivalent of H323, SIP and JXTA functions – these include establishing sessions, clients, profiles and a collection of shared resources. There is a new XML protocol XGSP introduced to capture the messaging needed to implement this capability, which we term "Collaboration as a Web Service". (http://grids.ucs.indiana.edu/ptliupages/publications/intl-sub03.pdf)
2) An event and messaging infrastructure NaradaBrokering (http://www.naradabrokering.org) that can manage the unicast and multicast delivery of messages between the different processes. NaradaBrokering copes with multiple protocols (both TCP/IP and UDP based) and tunnels through firewalls and network bottlenecks. It uses XML-based publish/subscribe semantics generalizing that familiar from JMS (Java Message Service) and we have shown this conveniently supports collaboration. Each shared object corresponds to a topic and NaradaBrokering manages the associated topic-labeled event streams with high performance.

We make the critical observation that Web services are built around messaging – their state is determined by control messages from the user or other services and their "meaning" (in particular their output display) is defined by messages sent from other Web services. This message-based structure allows both the universal implementation of collaboration and the convenient model for universal access. WSRP (Web Services for Remote Portals) is being developed by OASIS to specify the form of user-facing ports for Web services and WSIA (Web Services for Interactive Applications) and our approach unifies these.

Structure of SVG Web Service

The essential decomposition of SVG and related applications can be derived from the MVC picture. We take the *Model* component and this essentially becomes the Web Service while the *View* becomes the user interface. They are linked by the NaradaBrokering publish/subscribe messaging system; the combination of this with the preparation and interpretation of messages corresponds to the *Controller* MVC component. We analyze all possible events and divide them into DOM UIEvents (mouse and keyboard events) and semantic events (such as zooming). UIEvents are generated in the *View* and are converted into messages for the *Model*. One can design different *View* modules (with trade-offs in complexity and performance) through choice of which semantic events to process in the *Model* and which in the *View* component.

We support collaboration in two extremes; firstly the shared input port model where one replicates Web services and delivers events generated on a master *View* client to all instances of the *Model*. These service their associated *View* component. This has maximal flexibility for customization of each collaborative client while in the shared output port of service collaboration, a single *Model* instance uses NaradaBrokering to multicast rendering information to all collaborating *View* modules.

Structure of Collaborative W3C DOM/SVG Events

We define a collaborative event as an object that wraps original SVG events with additional context information for collaboration and Web service model. The context information helps to guide the events through the NaradaBrokering system to reach other clients (subscribers in the same session). The receivers un-wrap the collaborative event and get an SVG event that defines detailed actions on the SVG DOM. The *Model* part of Web service application analyses the SVG event based on its type and then delivers the resultant rendering information to the associated *View*(s). We will describe several types of events and give their structure. We distinguish change events and those that define the full application state; both are fully supported.

Conclusions

We believe our prototype shows how a message-based MVC model can generate a powerful application paradigm suitable for SVG and other presentation style applications. In the final paper, we will give architectural details and a detailed "before and after" performance analysis of the collaborative Web Service SVG application.