Project History

by Andrew C. Oliver

1. Brief Project History

The POI project was dreamed up back around April 2001, when Andy Oliver landed a short term contract to do Java-based reporting to Excel. He'd done this project a few times before and knew right where to look for the tools he needed. Ironically, the API he used to use had skyrocketed from around $300 ($US) to around $10K ($US). He figured it would take two people around six months to write an Excel port so he recommended the client fork out the $10K.

Around June 2001, Andy started thinking how great it would be to have an open source Java tool to do this and, while he had some spare time, he started on the project and learned about OLE 2 Compound Document Format. After hitting some real stumpers he realized he'd need help. He posted a message to his local Java User's Group (JUG) and asked if anyone else would be interested. He lucked out and the most talented Java programmer he'd ever met, Marc Johnson, joined the project. He ran rings around Andy at porting OLE 2 CDF and rewrote his skeletal code into a more sophisticated library. It took Marc a few iterations to get something they were happy with.

While Marc worked on that, Andy ported XLS to Java, based on Marc's library. Several users wrote in asking to read XLS (not just write as had originally been planned) and one user had special requests for a different use for POIFS. Before long, the project scope had tripled. POI 1.0 was released a month later than planned, but with far more features. Marc quickly wrote the serializer framework and HSSF Serializer in record time and Andy banged out more documentation and worked on making people aware of the project

Shortly before the release, POI was fortunate to come into contact with Nicola -Ken- Barrozzi who gave them samples for the HSSF Serializer and help uncover its unfortunate bugs (which were promptly fixed). More recently, Ken ported most of the POI project documentation to XML from Andy's crappy HTML docs he wrote with Star Office.

Around the same time as the release, Glen Stampoultzis joined the project. Glen was ticked off at Andy's flippant attitude towards adding graphing to HSSF. Glen got so ticked off he decided to grab a hammer and do it himself. Glen has already become an integral part of the

POI development community; his contributions to HSSF have already started making waves.

Somewhere in there we decided to finally submit the project to The Apache Cocoon Project, only to discover the project had outgrown fitting nicely into just Cocoon long ago. Furthermore, Andy started eyeing other projects he'd like to see POI functionality added to. So it was decided to donate the Serializers and Generators to Cocoon, other POI integration components to other projects, and the POI APIs would become part of Jakarta. It was a bumpy road but it looks like everything turned out since you're reading this!

2. What's next for POI

First we'll tackle this from a project standpoint: Well, we made an offer to Microsoft and Actuate (tongue in cheek ... well mostly) that we'd quit the project and retire if they'd simply write us each a really large check. I've yet to get a phone call or email so I'm assuming they're not going to pay us to go away.

Next, we've got some work to do here at Jakarta to finish integrating POI into the community. Furthermore, we're still transitioning the Serializer to Cocoon.

HSSF, during the 2.0 cycle, will undergo a few optimizations. We'll also be adding new features like a full implementation of Formulas and custom text formats. We're hoping to be able to generate smaller files by adding write-support for RK, MulRK and MulBlank records. I'm also going to work on a Cocoon 2 Generator. Currently, reading is not very efficient in HSSF. This is mainly because in order to write or modify, one needs to be able to update upstream pointers to downstream data. To do this you have to have everything between in memory. A Generator would allow SAX events to be processed instead. (This will be based on the low level structures). One of the great things about this is that, you'll not only have a more efficient way to read the file, you'll have a great way to use spreadsheets as XML data sources.

The HSSF Serializer, will further separate into a general framework for creating serializers for other formats and the HSSF Serializer specific implementation. (This is largely already true). We'll also be adding support for features already supported by HSSF (styles, fonts, text formats). We're hoping to add support for formulas during this cycle.

We're beginning to expand our scope yet again. If we could do all of this for XLS files, what about Doc files or PPT files? We're thinking that our next component (HDF - Horrible Document Format) should follow the same pattern. We're hoping that new blood will join the team and allow us to tackle this even faster (in part because POIFS is already finished). But maybe what we need most is you!

Copyright (c) @year@ The Apache Software Foundation All rights reserved. $Revision: 1.2 $ $Date: 2003/04/24 00:53:28 $