RCP Case Study: Tailored Technology for Every Customer

Introduction

Incremental Ltd. is small UK-based Company supplying production management software to SME's (small and medium sized enterprises) in niche markets within the engineering industry. Our current focus is ship repair.

We target the gap between commoditised, 'shrink wrapped' project management software and heavyweight ERP software suites. The former are cheap but generic, the latter powerful but expensive. Neither can satisfy both the requirements and budgets of SME's.

In 2003 we realised that the increasing ability of agile software development to rapidly and cheaply produce software, combined with the growing range of ever-more powerful open source technologies, could bridge this gap. We could produce software cheaply, and bringing in industry expertise, rapidly tailor it for individual niches, even individual customers. Once installed, we could then offer continuous improvement and customisation as our customers' needs evolved.

But how could we do this in a low-cost way, given that every niche, every customer, would have their own customised version of the product? We realised that the problem was not the development, but on-going support thereafter. To answer this, we turned to technology, to Eclipse and Eclipse RCP.

Architecture & Technology Selection

We started development in 2004. Our plans were ambitious – a multi-user, multi-currency, multi-lingual system that could be customised, and regularly updated, for each and every installation.

A web-based system seemed the obvious approach. However the user experience provided by web applications was, and is, simply not good enough for a system adopting a 'sovereign posture' [(ref 1)]– one which would be used for several hours each day. Indeed we had already seen a web-based predecessor fail in one niche market for exactly that reason. Hence we knew we must use some form of 'fat' client.

First, we looked at platform. Our market is world-wide, and cost conscious. Any constraints on deployment or licensing would work against us somewhere; hence Java was a straight-forward choice. Similar logic applied to the selection of database. We chose not to choose - our target persistence layer is the Hibernate O/R (we currently run against Oracle, Microsoft SQL Server, PostgreSQL and HSQLB).

We also had enough years' experience with behemoth J2EE application servers to know we would not be using one. We chose the Apache Avalon framework to host our server functionality.

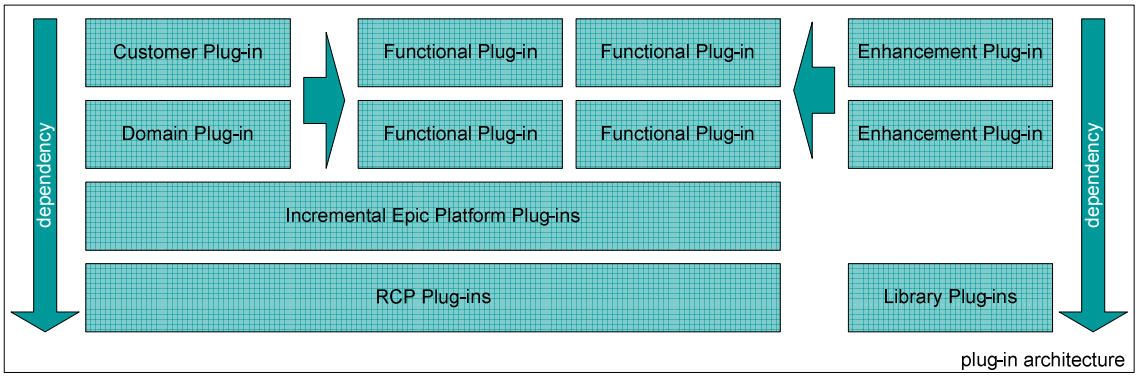
Next we had to choose our GUI technology. Java had already been chosen so it was a choice between Swing and SWT. SWT was a clear winner as it looked and performed better, but more importantly because it offered much better integration with the OS, particularly for our main target client platform, Windows XP. This was important as we knew had to integrate with office applications.

Thus we were going to develop an SWT application. Now we turned to other non-functional requirements. Whatever base system we produced had to be easily customisable, easily updateable, and with clean divisions between functional niches. Eclipse was already our

preferred development environment so we were familiar with the plug-in concept. It did not take us long to draw the obvious conclusion. This was early in 2004 and RCP had not been released; but Incremental Epic was born.

The Power of Plug-Ins

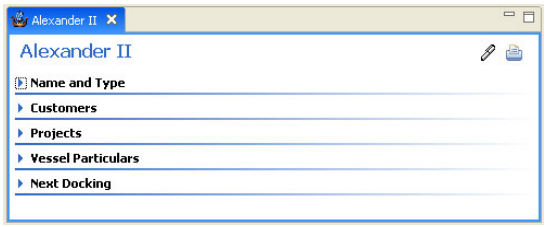
Our product, Incremental Epic, started as a set of Eclipse plug-ins with some gross hacks to disguise its IDE origins. In June 2004, to our relief, Eclipse 3.0 introduced RCP, and we switched Incremental Epic to a more comfortable base. However from the very start the plug-in concept permitted a clean architecture.

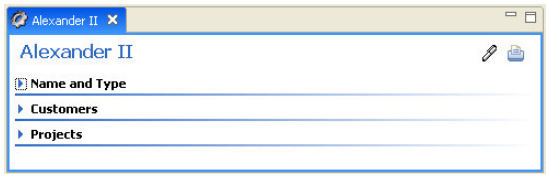


To the RCP platform, we added our own platform plug-ins providing abstract implementations of common elements (such as enhancements to the Forms GUI), a security model based upon JAAS, and most significantly our generic business entity model.

Upon this lie each our functional plug-ins, each representing a particular area within contract control – estimating, planning, purchasing, personnel, etc. Some, such as invoicing, rely on other functional plug-ins; others require only the platform.

The functional plug-ins are then tailored through a combination of enhancement and specialisation plug-ins. Enhancement plug-ins host functionality orthogonal to primary business logic such as reporting (based around BIRT) and Microsoft Office™ integration. The specialisation plug-ins are at domain (i.e. industry) and customer level. Through careful use of type and the extension point model, they are capable of specialising the business model and its presentation as well as offering customer-specific extensions.





the same record displayed with/without specialisation

Hence the plug-in model permitted us a large number of variations in our core offering whilst preventing an explosion in the number of development streams. Allied with such simple Eclipse concepts as working sets, we can manage several disparate customer environments with little more effort than one. The plug-in update mechanism allows equally straightforward distribution.

The Future

An unexpected boon arose from building our RCP product in an RCP product – the Eclipse IDE. We have found it straightforward to build in functionality that would otherwise we would never have attempted. For instance Incremental Epic has its own self-test framework that can be fired off by an administrative user. Based on Eclipse's own use of JUnit, this stresses both GUI and non-GUI behaviour. It has been invaluable in persuading customers' system administrators to move to the 'frequent update' mentality.

We are now considering how we can take further advantage of Eclipse's capabilities to make the lifetime management of our product even easier. From humble roots in Ant-based automated builds, we are now looking at such ideas as stretching continuous integration all the way to customer sites, and automating remote debugging.

Thus Eclipse and Eclipse RCP will continue be the main technological enabler for our company.

Prepared by Mike Evans, Technical Director of Incremental Inc.

References

1 : 'Your Program's Posture' – Alan Cooper www.cooper.com/articles/art_your_programs_posture.htm

Acronyms & Mentioned Technologies

	Apache Avalon	http://avalon.apache.org (closed)
BIRT	Business Intelligence and Reporting Tools	www.eclipse.org/birt/phoenix
	Eclipse	www.eclipse.org
ERP	Enterprise Resource Planning	-
GUI	Graphical User Interface	-
	Hibernate	www.hibernate.org
	HSQLDB	www.hsqldb.org
IDE	Integrated Development Environment	-
JAAS	Java Authentication and Authorisation Service	http://java.sun.com/products/jaas
	JUnit	www.junit.org
	MS SQL Server	www.microsoft.com/sql
	Oracle	www.oracle.com
O/R	Object Relational	-
OS	Operating System	-
	PostgreSQL	www.postgresql.org
RCP	Rich Client Platform	www.eclipse.org/home/categories/rcp.php
SME	Small & Medium sized Enterprises	-
SWT	Standard Widget Toolkit	www.eclipse.org/swt