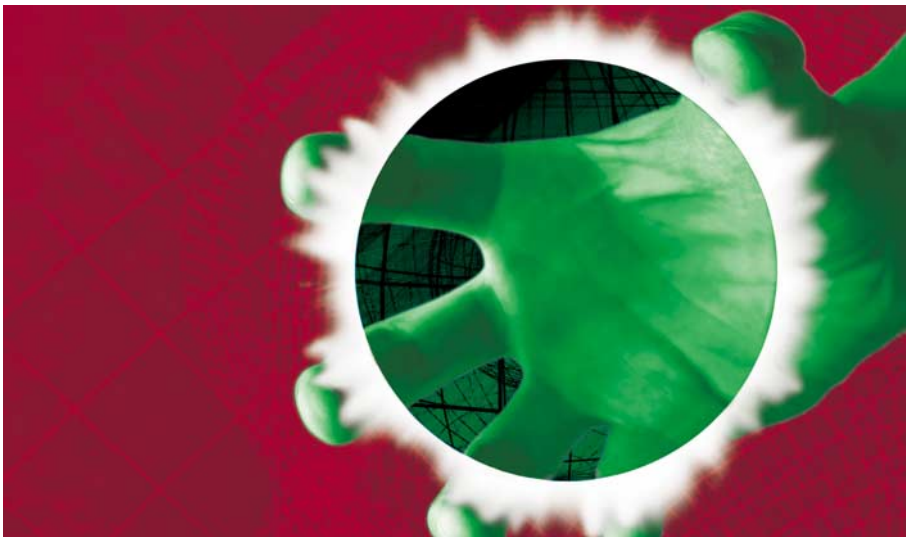
Neuigkeiten und Trends rund um die Eclipse Rich Client Platform

Wohin geht die Reise?

VON MARTIN LIPPERT UND BERND KOLB

Längst hat sich Eclipse von einer Plattform zur IDE-Entwicklung und -Integration zu einem allgemeinen Application-Framework gemausert. Seit mit der Version 3.0 die Eclipse Rich Client Platform veröffentlicht wurde, beginnen Entwicklungsorganisationen damit, ihre Rich-Client-Anwendungen auf Basis der Eclipse RCP zu implementieren. Da die Plattform noch recht jung ist, stellt sich zu Recht die Frage, welchen Reifegrad die Plattform bereits besitzt und wohin sie sich entwickeln wird. Vor allem der zweiten Frage wollen wir in diesem Artikel nachgehen.



Die Entwicklung der Eclipse Rich Client Platform ist mittlerweile zu einem bedeutenden Thema der Eclipse-Community geworden. Das bestätigen die diversen Vorträge, Tutorials und BoF-(Birds of a Feather-)Sessions auf der gerade vergangenen EclipseCon (siehe auch Bericht zur EclipseCon auf Seite 15), bei denen nicht nur technische Themen diskutiert, sondern auch im Einsatz befindliche RCP-Anwendungen demonstriert wurden.

Allerdings scheint die Entwicklung der Rich Client Platform selbst noch keineswegs am Ende angekommen zu sein. Ganz im Gegenteil. Wir werfen in diesem Artikel einen Blick auf die diversen Entwicklungen im RCP-Umfeld. Wer eine technisch detaillierte Diskussion oder Anleitung einzelner Themen sucht, sei dabei auf entsprechende (ggf. zukünftige) Spezial-Artikel verwiesen.

Neuigkeiten von der Plattform

Die Eclipse Rich Client Platform ist historisch aus der Eclipse Platform entstanden, indem die IDE-spezifischen Teile in separate Komponenten herausgelöst wurden. Allgemeine Konzepte wie Editors, Views, Perspectives, Workbench sowie Plug-ins können seitdem mit der Eclipse RCP verwendet werden, um allgemeine Rich-Client-Anwendungen zu realisieren. Allerdings ist diese historische Wurzel der Eclipse RCP durchaus anzumerken. Anforderungen, die für generelle Business-Anwendungen alltäglich sind, fanden in der Eclipse RCP teilweise (noch) keine Beachtung. Dazu zählen beispielsweise die direkte Integration in die Desktop-Umgebung, das möglichst einfache Starten einer solchen Anwendung mittels Java WebStart oder spezielle Anpassungen des Look & Feels an ein Corporate Design sowie ein Rechtemanagementkonzept.

Während einige dieser Anforderungen bereits in der Eclipse-Version 3.0 angegangen wurden, befinden sich andere noch in der Entwicklung oder können schon mit den aktuellen Milestone Releases der Version 3.1 genutzt werden. Sehen wir uns den aktuellen Stand an:

- Corporate Design: Mit der Version 3.0 hat Eclipse das Presentations API implementiert, mit dem sich unterschiedliche Darstellungen der Standardelemente innerhalb einer Workbench realisieren

lassen. Eclipse selbst bringt als beispielhafte Implementierung bereits unterschiedliche Presentations mit. Die aus der Version 3.0 üblichen rundlichen Darstellungen der Tabs lassen sich bspw. durch die aus der Version 2.0 bekannten eckigen Darstellung austauschen. Leider gibt es noch keine ausreichende Dokumentation zum Presentations API. Wer darüber mehr wissen möchte, kann sich aber die Implementierung der 2.1 Presentation in Eclipse 3.0 ansehen.

- Kiosk-Anwendungen: Eine interessante Möglichkeit, einen anderen Einstieg in eine RCP-Anwendung zu ermöglichen, bietet das Intro-Feature von Eclipse. Diese Seite, die alle Eclipse-Anwender vom ersten Start einer frischen Eclipse-Installation kennen, ist nichts anderes als eine DHTML-Seite. Das bedeutet nichts anderes, als dass sich nahezu beliebige DHTML-Inhalte auf der Intro-Seite unterbringen lassen. Z.B. lässt sich damit eine kioskartige Einstiegsoberfläche einfach realisieren und mit dem restlichen Unternehmens-Intranet verbinden. Wer also in seiner RCP-Oberfläche einen Film abspielen möchte: kein Problem. Jeff McAffer und Jean-Michel Lemieux haben dies auf der EclipseCon mit ihrer Beispielanwendung „Hyperbola“ (einem Instant Messaging Client) eindrucksvoll demonstriert.
- Das möglichst einfache und schnelle Deployment ist bei Rich-Client-Anwendungen ein wichtiges Thema. Umso mehr freut es, dass es seit der Eclipse-Version 3.1 M5a möglich ist, Eclipse-RCP-Anwendungen mittels Java WebStart zu starten. Entsprechende Wizards sind seit dem Milestone 6 komplett, um die passende JNLP-Site zu generieren. Darüber hinaus hat Ed Burnette auf der EclipseCon demonstriert, dass sich auch Windows-übliche Installer auf Basis von NSIS recht einfach für Eclipse-RCP-Anwendungen erzeugen lassen.
- Einige Teile des nativen Desktops (z.B. Windows) lassen sich bereits heute mit der Eclipse RCP ansprechen. Beispielsweise lassen sich Icons und Menüs in den System-Tray einhängen. Diese Features erreichen allerdings noch lange nicht die gewünschte Integrationstiefe mit dem nativen Desktop, die für manche Rich Clients wünschenswert wären. Ein neues Projekt-Proposal addressiert genau diesen Punkt: das Eclipse Desktop Technology Integration Project (www.eclipse.org/proposals/eclipse-desk/).
- Security: Im Equinox-Project von Eclipse wird derzeit daran gearbeitet, die Security-Features von Java auch in der Eclipse Runtime nutzen zu können. Damit können in Zukunft auch RCP-Anwendungen von den normalen Java 2 Security Features profitieren.
- Ein häufiger Kritikpunkt der Eclipse RCP ist, dass es einen erheblichen Aufwand bedeutet, die Plattform zu verstehen und nutzen zu können. Und in der Tat ist die Anzahl an Konzepten und APIs beachtlich. In den kommenden Versionen von Eclipse werden deshalb weitere Wizards und Templates realisiert werden. Mit ihnen soll es deutlich einfacher werden, die ersten Schritte zu einer Eclipse-RCP-Anwendung zu gehen. Im aktuellen Milestone Release von Eclipse 3.1 kann man bereits mit einem einfachen Template eine komplette kleine Mail-RCP-Anwendung generieren lassen. Weitere Wizards erleichtern das Branding der Anwendung (siehe Beitrag auf Seite 31).
- Bei dem bisher nicht immer ganz einfachen Deployment einer RCP-Anwendung wird man seit Eclipse 3.1 M5a ebenfalls unterstützt. Es gibt einen Wizard, welcher eine lauffähige Anwendung exportiert und sogar das Icon des Native Launcher Executable anpasst.
- Dass die Plug-in Runtime der Plattform nicht nur Vorteile bietet, zeigt sich bei RCP-Anwendungen immer dann, wenn externe Libraries eingebunden werden sollen. Häufig harmoniert das spezialisierte Classloading der Plug-in Runtime nicht mit solchen Libraries. Sie gehen meist davon aus, dass alle Klassen der kompletten Anwendung in einem einzigen gemeinsamen Classpath liegen. Eine Annahme, die bei der Plug-in Runtime von Eclipse so nicht mehr zutrifft. In der Regel lassen sich die auftretenden Probleme durch Work-arounds lösen [2]. Trotzdem müssen immer wieder neu die Ursachen eines Problems analysiert sowie eine passende Lösung implementiert werden. Das kommende Eclipse-Release soll dieser Problematik Rechnung tragen, indem einerseits die typischen Probleme inklusive deren Lösungen zentral dokumentiert werden. Andererseits sollen Hilfsklassen für Standard-Probleme geliefert werden, wo dieses möglich ist.

Libraries

Heute möchte niemand mehr seine Anwendungen komplett selbst implementieren. Für div. Aufgaben existieren ausgereifte Frameworks und Bibliotheken, die Open Source sind und sich einfach nutzen lassen. Auch im Eclipse-Umfeld entstehen immer mehr solcher Bibliotheken, die zur Implementierung von Rich-Client-Anwendungen nützlich sein können. Auch die Eclipse Foundation unterstützt die Entwicklung einer ganzen Reihe von Bibliotheken.

- Zu den bekanntesten Bibliotheken für RCP-Anwendungen gehört das Graphical Editor Framework (GEF, www.eclipse.org/gef/). Es bietet ein umfangreiches Framework, um grafische Editoren zu implementieren und befindet sich bereits in einigen kommerziellen Projekten im Einsatz. Es folgt dabei streng dem Model View Controller Pattern und bietet so die Möglichkeit einer exakten Trennung von Modell und Anzeige.
- EMF (Eclipse Modeling Framework, www.eclipse.org/emf/): Das EMF ist ein Framework, welches ein Meta-Metamodell und die notwendige Infrastruktur zur Verfügung stellt, um modellgetriebene Softwareentwicklung zu unterstützen und zu ermöglichen. Mittlerweile existieren bereits einige Projekte, welche auf EMF-Modellen arbeiten und z.B. Modellvalidierungen und Modell-zu-Modell-Transformationen ermöglichen.
- VE (www.eclipse.org/vep/): Bei VE handelt es sich um ein Framework zur Erstellung von GUI-Buildern. Es enthhält Referenzimplementierungen für Swing und SWT. Dabei nutzt das Projekt sowohl GEF als auch EMF.

Neben diesen ausgewählten Projekten entstehen derzeit eine Reihe weiterer Projekte, die für RCP-Anwendungen interessant werden können. Dazu zählen sicherlich das Eclipse Communication Framework, das TPTP-Projekt, das Model-Driven-De-

velopment- Integration-Projekt oder auch das BIRT-Projekte.

RCP everywhere

Mithilfe der Eclipse Rich Client Platform entwickelte Anwendungen lassen sich zunächst einmal nur auf den Plattformen einsetzen, für die die Eclipse-Plattform zur Verfügung steht. Neben der Windows-Plattform sind dies derzeit: Linux, Solaris 8, AIX, HP-UX und Mac OS X. Voraussetzung dafür, dass eine RCP-Anwendung auf allen diesen Plattformen läuft, ist natürlich, dass in der RCP-Anwendung keine plattformspezifischen Features implementiert wurden (etwa durch die COM/ActiveX Bridge oder JNI).

Über diese „klassischen" Client-Plattformen hinaus werden zunehmend kleine Plattformen für Client-Anwendungen interessant, beispielsweise Handheld Devices wie PocketPC- oder Palm OS-Geräte oder Handys, die schon heute eine Java-VM installiert haben. Besonders herausfordernd bei der Entwicklung von Anwendungen für diese Geräte ist es, trotz der beschränkten Möglichkeiten ein möglichst ähnliches Programmiermodell zu realisieren, wie für normale Rich-Client-Anwendungen. Dies ist dann besonders interessant, wenn ein Teil einer normalen Desktop-Rich-Client-Anwendung auch auf mobilen Devices zur Verfügung stehen soll. Je mehr Code sich die mobile Anwendung und die Desktop-Anwendung teilen können, desto geringer wird der Aufwand, beide Frontends zu entwickeln.

Die Rich Client Platform von Eclipse ist zugeschnitten auf Desktop-Rich-Client-Anwendungen und lässt sich so ohne weiteres nicht auf mobilen Geräten verwenden. Deshalb existiert ein spezielles Eclipse-Projekt, welches sich zum Ziel gesetzt hat, eine Rich Client Platform für Embedded Devices zu realisieren: Das eRCP Project (www.eclipse.org/ercp/). Innerhalb dieses Projektes werden unter anderem Versionen von SWT, JFace, der Runtime, der Workbench und des Update-Me- chanismus für Embedded Devices implementiert. Dabei werden als Target-Plattformen zunächst PocketPC und Nokia Series 80-Plattform realisiert. Auf der EclipseCon konnte man bereits eine RCP-Anwendung auf einem Nokia 9500 bewundern.

Technisch wird eRCP ein Subset der normalen RCP darstellen, was einerseits bedeutet, dass natürlich nicht alle RCP-Anwendungen „einfach so" auch auf Embedded Devices funktionieren werden. Fraglich ist dabei auch, ob dies überhaupt sinnvoll wäre, da Embedded Devices in der Regel starken Beschränkungen der Bildschirmgröße und der Interaktionsmöglichkeiten unterliegen. Es zeigt allerdings, dass sich RCP-Anwendungen mit dem gleichen Programmiermodell für normale Desktop-Systeme und für Embedded Devices implementieren lassen. Modularisiert man die RCP-Anwendung darüber hinaus in sinnvolle Plug-in-Einheiten, können große Teile einer Anwendung auf beiden Plattformen genutzt werden. Der Trick besteht dann darin, die UI- Teile einer Anwendung sauber von den nicht-UI-Teilen zu trennen und die UI-Teile speziell für die Devices in unterschiedlichen Plug-ins zu entwickeln.

Rich Application Server Platform

Die Eclipse Rich Client Platform ist, wie der Name schon sagt, auf Client-Anwendungen fokussiert. Das ist auch deshalb nicht verwunderlich, da die Rich Client Platform hauptsächlich aus UI-Frameworks besteht (SWT, JFace, Workbench). Dennoch finden sich immer mehr Entwickler, die die Basis-Runtime einer RCP-Anwendung (inkl. der anwendungsspezifischen Nicht-UI-Plug-ins) gerne für die serverbasierten Teile einer Business-Anwendung nutzen würden. Verständlich angesichts der Tatsache, dass heute kaum eine Business-Anwendung ausschließlich als Fat Client implementiert wird.

Es stellt sich also die Frage, inwiefern man auch den serverbasierten Anteil einer Business-Anwendung auf Basis der Eclipse-Plattform implementieren kann, um auch dort den Plug-in- und den Extension-Point-Mechanismus zu nutzen. Während es zunächst natürlich ohne weitere Probleme möglich ist, eine Nicht-UI-Anwendung auf Basis der Eclipse Runtime zu starten (sog. Headless-Applikationen), gestaltet sich dieser Einsatz schwieriger, wenn der serverbasierte Anteil einer Anwendung innerhalb eines Web- oder Application Server ablaufen soll. In der Regel muss dann ein spezielles Startup-Skript implementiert werden, mit dem das Classloading der Eclipse Runtime und das Classloading des Web- oder Application Server miteinander in Einklang gebracht werden. In den entsprechenden Newsgroups findet man dazu eine Reihe von Postings und auch Gespräche auf der EclipseCon verraten, dass es möglich und sinnvoll ist.

Und tatsächlich taucht schon in dem März-Newsletter von Eclipse Source (www.eclipsenews.com) ein kurzer Abschnitt auf, in dem über eine so genannte Eclipse Application Server Platform (RASP) gesprochen wird. Obwohl der News-Eintrag recht nebulös bleibt, scheint es doch so, als ob sich in diesem Bereich etwas bewegt.

Resumee

Die Eclipse Rich Client Platform hat sich zu einem ernst zu nehmenden Bestandteil der Eclipse Platform etabliert. Und die aktuelle Entwicklung zeigt, dass das Eclipse-Projekt die Weiterentwicklung der RCP vorantreibt – sowohl im Eclipse-Platform-Projekt selbst als auch flankiert um weitere Projekte. Die im Einsatz befindlichen RCP-Anwendungen zeigen, dass sich mit der Rich Client Platform eindrucksvolle Anwendungen entwickeln lassen (www.eclipse.org/community/rcp.html).

Martin Lippert ist Senior-IT-Berater bei it-agile. Er arbeitet dort als Coach und Berater für agile Softwareentwicklung, Refactoring und Eclipse-Technologie. Kontakt: martin.lippert@it-agile.de.

Bernd Kolb ist freiberuflicher Berater und Coach. Er ist (Mit-)Autor von diversen Artikeln sowie eines Buches und Sprecher auf Konferenzen. Seine Schwerpunkte liegen auf modellgetriebener Software- entwicklung sowie Eclipse. Kontakt: b.kolb@kolbware.de.

Links & Literatur

[1] Eclipse Rich Client Platform: www.eclipse.org/rcp/. Hier finden sich viele nützliche Links, Tutorials und Beispiele zur Eclipse RCP.

[2] Martin Lippert: Classloading in Eclipse, in JavaSpektrum 1.2005. Der Artikel beschreibt die typischen Tücken mit dem speziellen Classloading in Eclipse und gibt Tipps und Tricks, wie mit diesen Tücken umgegangen werden kann. Siehe auch www.martinlippert.org

[3] Boris Bokowski, Frank Gerhardt: Von der Stange. Maßgeschneiderte grafische Editoren mit dem Graphical Editing Framework, in *Eclipse Magazin* Vol. 2

[4] Jens von Pilgrim: Mr. M. Agile MDA mit EMF, in *Eclipse Magazin* Vol. 2