Capitolul 1

Tehnologii

1.1 Eclipse Rich Client Platform

Proiectul Eclipse îşi are începuturile alături de compania IBM de care s-a separat în 2003; astăzi este printre cele mai cunoscute proiecte opensource, fiind recunoscut ca un foarte popular mediu de dezvoltare de aplicaţii Java şi nu numai.

Binecunoscutul mediu de dezvoltare Eclipse este dezvoltat pe baza Eclipse Rich Client Platform. Aceasta reprezintă suita minimă de plugin-uri necesare pentru dezvoltarea unei aplicaţii "rich client" [1]. Deşi platforma este destinată dezvoltării de aplicaţii tip IDE, prin înlăturarea anumitor componente ale platformei se pot dezvolta orice tip de aplicaţii desktop.

Structura de bază a acestei platforme poate fi rezumată după cum am prezentat în Tabela 1.1.

1.1.1 Eclipse Runtime

Această componentă a Eclipse RCP este cea mai abstractă şi în acelaşi timp cea mai puternică componentă a sa. Structura sa de bază este descrisă în Tabela 1.2.

Equinox este o implementare a specificaţiilor OSGi R4 [2] o serie de plugin-uri care implementează diverse servicii opţionale OSGi şi alte infrastructuri pentru rularea sistemelor bazate pe OSGi. [3]

Equinox produce, printre alte lucruri, implementarea OSGi folosită de Eclipse RCP (şi celelalte proiecte bazate pe Eclipse) ca un model de componente. [2]

[1] eng. *client bogat* – O aplicaţie desktop care beneficiază de un set de elemente de interfaţă evoluate ("bogate"), care oferă o putere mare de abstractizare şi în acelaşi timp control al funcţionalităţii programatorului, păstrînd o experienţă de utilizare plăcută pentru utilizator. În general orice aplicaţie desktop care foloseşte elemente de interfaţă predefinite pînă la un anumit punct poate fi considerată ca un rich client.

[2] `http://osgi.org/osgi_technology/download_specs.asp?section=2#Release4`

Tabela 1.1: Structura Eclipse RCP [6]

Eclipse Runtime	Suport pentru plugin-uri, puncte de extensie şi extensii. Construită peste framework-ul OSGi
SWT	Proiectat să ofere acces eficient şi portabil la facilităţile de interfaţă utilizator ale sistemului de operare pe care este implementat
JFace	Un framework de interfaţă utilizator, construit pest SWT, pentru tratarea multor sarcini de programare de interfeţe
Workbench	Construit peste toate componentele anterioare, aduce un mediu de lucru ultra scalabil, cu interfaţă publică ce suportă mai multe ferestre pentru administrarea vizualizărilor, editoarelor, perspectivelor, acţiunilor, preferinţelor şi multe altele.

Tabela 1.2: Structura Eclipse Runtime [1]

org.eclipse.core.contenttype	Suport pentru definirea şi administrarea tipurilor de fişiere
org.eclipse.core.jobs	Infrastructură pentru programare paralelă în Eclipse
org.eclipse.equinox.registry	Sistemul prin care un plugin publică alte plugin-uri de care depinde şi defineşte puncte de extensie pentru ca alte plugin-uri să-i îmbogăţească funcţionalitatea
org.eclipse.equinox.preferences	O infrastructură prin care un plugin îşi păstrează preferinţele în seturi de perechi cheie/valoare modificabile de către utilizator
org.eclipse.equinox.common	Administrarea proiectelor, resurselor şi fişierelor

1.1.2 SWT

SWT[3] este una din cele mai populare tehnologii dezvoltate de Fundaţia Eclipse. Această tehnologie foloseşte la crearea de interfeţe utilizator în mediul de dezvoltare Java, oferind programatorului capacitatea de a construi interfeţe utilizator portabile între diverse sisteme de operare. În prezent suportă toate sistemele de operare uzuale şi diverse arhitecturi maşină (e.g. x86, amd64).

Structura sa este similară cu cea a altor toolkit-uri similare, ca AWT [4] şi SWING[5]. Elementele fundamentale sunt Shell-ul şi Controlul. Un shell este abstracţia conceptului de fereastră şi reprezintă în general un container pentru controale. Controalele sunt elementele interfeţei utilizator.

[3]abrev. *Standard Widget Toolkit*

[4]abrev. *Abstract Window Toolkit* – toolkit-ul IU standard Java `http://en.wikipedia.org/wiki/Abstract_Windowing_Toolkit`

[5]Un toolkit IU independent de platformă `http://en.wikipedia.org/wiki/Swing_(Java)`

Butoanele, etichetele, arborii şi tabelele sunt toate controale şi utilizatorii sunt obişnuite cu ele din alte programe ale desktopului. [8] În principiu orice widget poate fi simplu sau compus (i.e. descendent al clasei Composite) şi întreaga lor ierharhie este controlată de relaţiile dintre clasele SWT.

Interacţiunile dintre controale şi utilizator cît şi controlul tranziţiei stărilor controalelor se face cu ajutorul Evenimentelor. Fiecare modificare de stare a unui control (cum ar fi apăsarea pe un buton) este semnalată de SWT către aplicaţie prin declanşarea unui eveniment. Orice Listener (ascultător înregistrat pentru un anumit eveniment) va primi o notificare (i.e. o metodă declarată în interfaţa de Listener va fi apelată) ce va conţine evenimentul ce a avut loc. Orice clasă poate deveni listener prin implementarea unor interfeţe specifice, SWT oferind astfel o mare flexibilitate în cadrul aplicaţiilor care îl folosesc.

Ca şi în AWT sau SWING, poziţia controalelor este stabilită cu ajutorul Layout-urilor. Un layout reprezintă un set de reguli de poziţionare pentru un control compus ce va decide pentru toate controalele incluse cum vor fi ele poziţionate pe ecran. Unele layout-uri asociază anumite date fiecărui control pentru a reţine poziţia în care va fi desenat. Poziţionarea manuală a tuturor controalelor este o problemă complexă şi dacă acel algoritm nu este scris eficient, el poate afecta performaneţele întregii aplicaţii. De aceea, SWT oferă nişte seturi predefinite de layout-uri ce serversc cele mai comune cazuri întîlnite de programatori.

1.1.3 JFace

JFace este un toolkit pentru interfeţe utilizator ce conţine clase pentru tratarea multora din sarcinile uzuale de programarea interfeţelor utilizator. JFace este independent de sistemul de ferestre atît in interfaţa sa pentru programatori cît şi în implementare, şi este proiectat să funcţioneze cu SWT fără a-i ascunde funcţionalitatea.

JFace include componentele uzuale de toolkit IU, regiştri de imagine şi seturi de caractere, text, dialog-uri, framework-uri pentru preferinţe şi wizard-uri [6] şi raportarea progresului pentru sarcini cu durată mare de execuţie. [4]

Pachete principale din JFace, care oferă funcţionalitatea de bază a tehnologiei sunt prezentate în Tabela 1.3

1.1.4 Workbench

Această parte a Eclipse RCP reprezintă o colecţie largă de clase şi interfeţe pentru construirea de interfeţe utilizator complexe.

Iată principalele elemente ce constituie un Workbench.

[6] Dialogurile des întîlnite ce automatizează diverse sarcini repetitive

Tabela 1.3: Structura JFace [7]

Ferestre	org.eclipse.jface.window	Facilităţi de creerea şi administrarea ferestrelor. Interesantă este clasa ApplicationWindow, care aduce un nou nivel de abstracţie peste conceptul de fereastră şi înglobează o buclă program SWT.
Vizualizări	org.eclipse.jface.viewers	Vizualizări ca şi TreeViewer sau TableViewer, care sunt componente ghidate de model ce folosesc widgeturi SWT şi adaptează conţinutul modelului la conţinutul widgetului.
Dialog-uri	org.eclipse.jface.dialogs	Diverse dialog-uri des folosite.
Acţiuni	org.eclipse.jface.actions	Un framework de acţiuni IU similar cu cel găsit în SWING pentru a implementa comportament partajat între două sau mai multe componente, cum ar fi o intrare în meniu şi un buton de pe bara de instrumente.
Wizard-uri	org.eclipse.jface.wizard	Un framework avansat de construcţie a wizard-urilor.
Resurse	org.eclipse.jface.resource	Suport pentru administrarea resurselor, cum ar fi imaginile şi fonturile SWT.
Text	org.eclipse.jface.text	Un framework de crearea, manipularea, afişarea şi editarea documentelor text.

Workbench

Ca termen strict, se referă la fereastra care conţine întreaga aplicaţie. Defineşte un container abstract pentru toate elementele de interfaţă din aplicaţia dezvoltată cu Eclipse RCP.

Pagina

Reprezintă, în mod intuitiv, toată partea ferestrei care nu conţine barele de unelte şi meniul.

Perspective

Folosesc pentru organizarea conţinutului într-o Pagină. O perspectivă defineşte o colecţie de Vizualizări, aşezarea lor şi acţiunile ce pot fi efectuate pentru o sarcină a utilizatorului. Perspectivele pot fi schimbate de către utilizatorul. Modificarea Perspectivei afectează doar Vizualizările nu şi Editoarele. [1]

Vizualizările şi Editoarele

Reprezintă extensiile aduse de programator într-o aplicaţie Eclipse RCP. Un editor urmăreşte modelul unei maşini de stări în sensul că el are o etapă de deschidere, editare şi o etapă de salvare.

În schimb orice modificare a stării unei Vizualizări este salvată imediat. Vizualizările sunt folosite la a arăta structura modelului deschis într-un Editor, la a facilita accesul la fişiere şi la alte resurse ale unui proiect.

Editoarele urmăresc în general modelul unui editor de text clasic şi oferă aceeaşi funcţionalitate însă la un nivel mai abstract, nefiind dependent de o intrare ca fişier text, nici măcar ca intrarea să fie orice fel de fişier.

1.2 OpenGL şi extensia OpenGL pentru SWT

OpenGL[7] este o specificaţie de standard independent de limbaj ce defineşte un API pentru scrierea de aplicaţii ce produc grafice tridimensionale sau bidimensionale pe calculator. Interfaţa este compusă din peste 250 de apeluri de funţii ce pot fi folosite pentru a desena scene tridimensionale complexe din primitive simple. OpenGL a fost dezvoltat de SGI[8] în 1992. Este folosit pe scară largă în aplicaţii CAD, realitate virtuală, vizualizare ştiinţifică, vizualizarea informaţiei, simularea zborului, dezvoltare de jocuri pe calculator, ş.a. [5]

[7]abrev. *Open Graphics Library*
[8]abrev. *Silicon Graphics Inc.*

Bibliografie

[1] Eclipse sdk documentation
http://help.eclipse.org/.

[2] Equinox frequently asked questions
http://www.eclipse.org/equinox/faq.php.

[3] Equinox
http://www.eclipse.org/equinox/.

[4] Jface
http://wiki.eclipse.org/index.php/JFace.

[5] Opengl on wikipedia
http://en.wikipedia.org/wiki/OpenGL.

[6] Martin Oberhuber, Nick Boldt, Daniel Spiewak, Thomas Bonk, et al. Rich client platform frequently asked questions
http://wiki.eclipse.org/index.php/RCP_FAQ.

[7] Brian Sam-bodden and Christopher Judd. Rich clients with the swt and jface
http://www.javaworld.com/javaworld/jw-04-2004/jw-0426-swtjface.html.

[8] Joe Winchester and Steve Northover. Swt - a native widget toolkit for java
http://java.sys-con.com/read/37463.htm.