Gliederung

- Vorstellung der Teilnehmer
- Beschreibung von Freeciv
- Bewertung von Freeciv
- Diskussion
- Praxis

Vorstellen der Teilnehmer

Workshop?

- einen Workshop zeichnet aus
  - klare Zielsetzung
  - Ergebnis-Offenheit
  - geeignete Arbeitssituation
  - aktive Mitarbeit
- nicht ganz zutreffend, da kein wirkliches Ziel
- Ziel soll sein, die Teilnehmer in die Lage zu versetzen selbst ein Open-Source Spiel zu starten

Freeciv

- www.freeciv.org
- multi-player netzwerkfähgiges rundenbasierendes Strategiespiel
- erste Version 1995, jetzt stabil und ausgereift
- Ports für: Unix, Windows, Amiga, VMS (PDAs)
- Clients für: Xaw, GTK+ (1 und 2), Win32, MUI und SDL
- Geschrieben in C
- minimale Abhängigkeiten: UNIX ähnliche Umgebung (Netzwerk, Shell), ANSI-C Compiler, make + Client Abhängigkeiten

Technische Details

- single-threaded Server
- TCP/IP Verbindungen zwischen Client und Server
- Serialisierung von Hand
- Protokoll ist versioniert und sub-versioniert
- Protokoll ist request $\rightarrow$ response
- Servermeldungen nur in einer Sprache
- AI ist fest im Server eingebaut
- dumb client bekommt alle Daten von Server
- Client ist allein für Darstellung verantwortlich
- Tilesets
- I18N für 21 Sprachen

Hosting Details

- eigene Domain
- eigene Hardware
- Bandbreite von Arbeitgeber gestellt
- 2 Administratoren (programmiereren nicht)
- Kommunikation über: Mailinglisten, irc, USENET Gruppe (selten)
- RT als Requesttracker mit 800+ resolved und 250+ open issues

Administrative Details

- 7 Maintainer mit CVS Schreibrechten
- niemand darf eigene Patches committen
- Patches haben eine Haltezeit für Review
- Konsensus bei Entscheidungen == jeder hat Veto
- 2 Client-Maintainer
- dürfen in eigenen Client Verzeichnis sofort und selbst schreiben

Legale Details

- die Maintainer achten darauf, daß nicht Material aus fragwürdigen Quellen verwendet wird
- kein Problem bei Code aber bei Grafik, Sound und Text
- bisher keine Probleme

Bewertung von Freeciv

- Aufzeigen von Schwächen und Stärken
- persönliche Einschätzung
- Stabil und ausgewachsen
- wenig Änderungen notwendig per se

Architektur (1)

- Client-Server, single-threaded, TCP/IP, Versionierung und request $\rightarrow$ response sind gut
- Serialisierung von Hand ist schlecht
  - Portierung wird erschwert
  - bisher nur ein “externer” Client
  - in Java
  - nicht mehr gepflegt
  - Netzwerkprotokoll muss per Hand übertragen werden
  - Lösung: anderer Ansatz (CORBA, XMLRPC, SOAP, RPC, ...) oder maschienenlesbare Protokollbeschreibung mit verschiedenen Ausgabesprachen

Architektur (2)

- Protokoll ist nicht selbstklärend
  - zuviele Konstanten
  - zuviel Wissen notwendig auf Clientseite
  - Lösung: noch dümmerer Client
- Nachträglicher Einbau von Sicherheit in Protokolle
  - Freeciv unterstützt jetzt Authentisierung
  - Persistenz über mehrere Spiele notwendig
  - neues Interface (Info über Account, ...) notwendig
- Server sollte I18N-neutral sein
- AI sollte seperater Client sein
- Karteneditor
- Kartengenerator

Entscheidungsfindung

- Konsens unter 7 Personen ist schwierig
- Entscheidungen sind langwierig
- in der Geschichte von Freeciv auch Phasen von Diktatoren
- mit einzelner Executive schneller Entscheidungen aber eventuell “falscher” Weg
- Releaseanleitung vs. Releaseziele

Weitere Entwicklungen

- Architekturprobleme lösen
- Generalisierung
  - wie generell ist notwendig/sinnvoll
  - kosten meist Performance
  - Henne-Ei Problem mit Nutzeren der Generalisierungen
- neue Clienten, Clienten updaten (GTK1 -> GTK2)

Client und AI

- seit längerem serverseitige Auto-Features
- AI sollte als seperater Client ausgelagert werden
- Möglichkeit Teile der AI dem Spieler als Auto-Features zur Verfügung zu stellen
- Client muss Regeln des Spieles kennen
- Wiederspruch zu dumb client

Weitere Anmerkungen

- Spam über Mailingliste
- Spam über bug Report Addresse
- Mangel an künstlerisch begabten Mitwirkenden

Diskussion

Praxis