JavaPairing

gestione del torneo di scacchi

Convegno Nazionale Arbitri
Roma 6 febbraio 2010
(e successivi aggiornamenti)

FSI

Eugenio Cervesato,
AR

Curriculum scacchistico personale

Posizione F.S.I.: GIOCATORE DI CATEGORIA 3N (1995); ISTRUTTORE ELEMENTARE (2001); ARBITRO REGIONALE (2003). Co-fondatore e Presidente del Circolo Scacchi “Bobby Fischer” di Cordenons. Sono stato Presidente del Comitato Regionale del Friuli Venezia Giulia dal 2005 al 2008. Svolgo attività agonistica nel Campionato Italiano a Squadre.

Autore delle dispense: “CORSO DI SCACCHI PER PRINCIPIANTI Nozioni base per iniziare il gioco e tecniche combinative semplici” e “CORSO DI SCACCHI PER PRATICANTI DI PRIMO LIVELLO combinazioni elementari per affinare l'intuizione scacchistica”.

Dal 2000 a tutt'oggi Istruttore di ragazzi con lezioni a cadenza settimanale in sede.
Ho tenuto vari corsi di scacchi per principianti nelle scuole.

Ho organizzato vari tornei: studenteschi, GSS, giovanili, semilampo individuali e a squadre, sociali. Ho collaborato con l’arbitro principale in tornei validi per Elo.

Autore del programma ITSV per abbinamento italosvizzero (anni 1999-2003), scritto in Basic per DOS.

Autore del programma di gestione del CIS-FVG (2007), scritto in php/HTML, per girone all’italiana.

Ho aggiornato il programma ‘calcolatore Elo’, scritto in JavaScript, alle specifiche FIDE in vigore dal 1.7.09

Conoscenze professionali

- Sistemi operativi: DOS, Windows, Linux
- Linguaggi di programmazione: Fortran, Basic, Clipper, Object Pascal, C++, Java, php, Javascript, HTML
- Sistemi database: DBase, Interbase, Firefox, Oracle
- Applicazioni desktop e Office
- Internet / Intranet – design siti WEB
- Inglese tecnico-scientifico

Conoscenze scacchistiche + conoscenze professionali

Perché sviluppare un nuovo programma per la gestione del torneo di scacchi?

ho utilizzato vari programmi: Diena, GTS, Sevilla e Vega/VegaTeam (nel periodo 2007-2008 ho collaborato con Luigi Forlano al test di VegaTeam versione Linux). Comunque tali software presentano limitazioni che ritengo esagerate e possono nascondere errori di programmazione impossibili da scoprire in quanto il codice sorgente non viene reso pubblico (questo crea problemi anche alla FIDE).

Volevo un programma unico per tornei a squadre ed individuali, un prodotto snello, che facilitasse il lavoro dell'arbitro durante il torneo, per esempio: consentire di inserire anche i ritardatari dal database e sostituire i titolari con le riserve direttamente all'atto dell'inserimento dei risultati.

Volevo un programma che consentisse tornei regolari ma anche a doppio girone o a girone completamente libero e che permettesse di modificare in corsa il sistema di abbinamento.

Infine volevo un programma cross-platform vero, in grado cioè di girare su tutte le piattaforme ed internazionalizzato.

E il senso della sfida con se stesso? (... da scacchista ...)

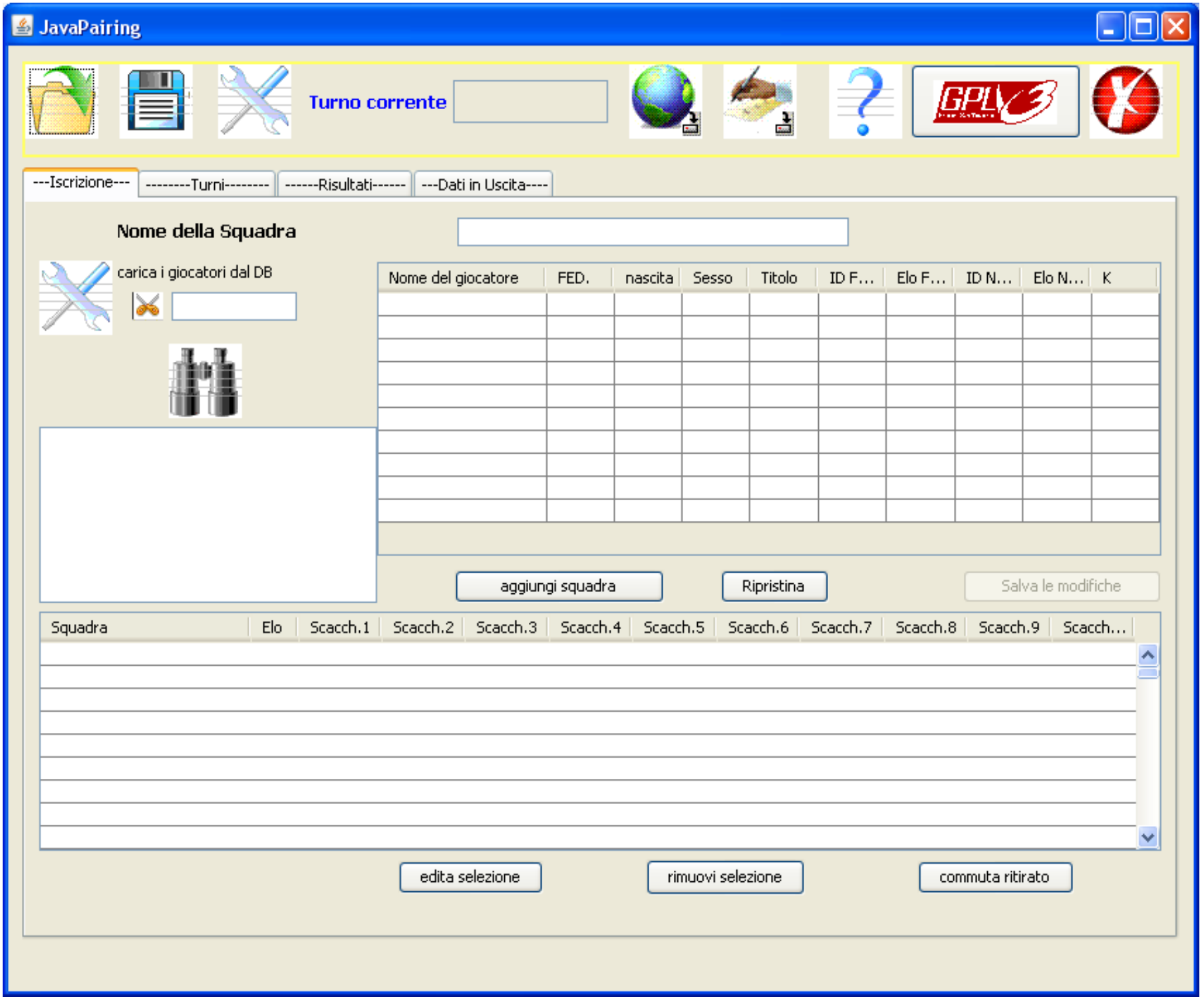
Java è cross-platform per definizione.

La programmazione in Java con editor di testo è una cosa pazzesca!!

Nel 2007 mi sono casualmente imbattuto nel prodotto NetBeans IDE della Sun Microsystems Inc, che finalmente consentiva il design interattivo dell'interfaccia grafica, la compilazione ed il debug integrato, come ero abituato a fare da anni con i prodotti della Borland. Ho tuttavia avuto necessità di un lungo periodo di training per riuscire a padroneggiare il design dell'interfaccia utente, in quanto ero abituato a specificare le coordinate in pixel, mentre in Java gli oggetti sono liberi di ridimensionarsi.

Da qui la decisione finale (dicembre 2008) di cimentarmi con lo sviluppo di JavaPairing, andando per step successivi

1. design dell'interfaccia utente
2. tabellone/classifica/variazione Elo
3. importazione dei dati dei giocatori
4. generazione del sito WEB e del file per l'omologazione
5. implementazione dei motori di abbinamento


JavaPairing
Turno corrente
GPL v3
---Iscrizione---
--------Turni--------
------Risultati------
---Dati in Uscita----
Nome della Squadra
carica i giocatori dal DB
Nome del giocatore
FED.
nascita
Sesso
Titolo
ID F...
Elo F...
ID N...
Elo N...
K
aggiungi squadra
Ripristina
Salva le modifiche
Squadra
Elo
Scacch.1
Scacch.2
Scacch.3
Scacch.4
Scacch.5
Scacch.6
Scacch.7
Scacch.8
Scacch.9
Scacch...
edita selezione
rimuovi selezione
commuta ritirato

... *repetita iuvant* ... La distribuzione gira su qualsiasi piattaforma così com'è, compresi Windows, Linux e Macintosh, è solamente richiesto il supporto Java (JRE 1.6 o successivo).

Per installare, espandi il file '.zip' nel disco fisso, conservando la struttura delle sottocartelle.
Esegui il file 'JavaPairing.jar' che si trova nella sottocartella 'dist' (in effetti questo è l'unico file che realmente serve ad eseguire il programma).

http://javapairing.sourceforge.net/ita

Implementazione in JavaPairing dell'algoritmo 'Svizzero Olandese'

- il numero di turni è stabilito in anticipo. (può essere esteso)
- due giocatori possono incontrarsi una sola volta (può essere esteso)
- Le partite con il bianco e con il nero dovrebbero essere pari e alternate. Sequenze di tre partite con lo stesso colore o differenze maggiori di 2 sono consentite solo all'ultimo turno per i classificati sopra il 50% se questo riduce i downfloaters. In caso di uguale colore atteso, viene valutata la priorità del colore, la storia e in ultima analisi prevale il giocatore meglio classificato.
- 1 punto per la vittoria, ½ per la patta, 0 per la sconfitta. Forfeit e BYE valgono un punto e la partita è considerata giocata senza avversario e colore (ai fini dello spareggio tecnico = patta contro se stesso)
- gli ID sono assegnati per Elo decrescente (può rendersi necessario un intervento manuale per integrare l'assegnazione per titolo o altri criteri)
- i ritardatari sono gestiti in modo dinamico, senza penalizzazioni
- se dispari, riposa il giocatore, che non abbia mai avuto punti senza giocare, rimasto spaiato alla fine dell'abbinamento
- a partire dalla testa della classifica parziale, si tenta di abbinare giocatori a pari punteggio. Se si forma un gruppo dispari, l'ultimo rimasto spaiato sarà poi mosso al gruppo successivo. ...

... quindi la prima metà (S1) viene appaiata alla seconda metà (S2), eseguendo delle permutazioni in S2 se la partita è già stata giocata o c'è un vincolo di colore o float (non è ammesso upfloat o downfloat per due volte consecutive o come due turni prima, eccetto all'ultimo turno per i classificati sopra il 50% se questo riduce i downfloaters).
Se risulta impossibile fare le coppie, si effettuano scambi tra S1 ed S2.
Se ancora risulta impossibile fare le coppie, i residui vengono mossi al gruppo successivo.
Se l'ultimo gruppo non può essere appaiato, si precede ad unirlo con il penultimo, spezzando le coppie già formate e ripetendo la procedura fino ad ottenere un abbinamento legale.

n.b. L'implementazione è pienamente aderente alle regole della FIDE
(per dettagli si veda l'Handbook 2010 della FIDE presente nella cartella 'swiss')

Implementazione in JavaPairing dell'algoritmo 'Svizzero Dubov'

- il numero di turni è stabilito in anticipo. (può essere esteso)
- due giocatori possono incontrarsi una sola volta (può essere esteso)
- Le partite con il bianco e con il nero dovrebbero essere pari e alternate. Sequenze di tre partite con lo stesso colore o differenze maggiori di 2 non sono consentite. In caso di storia di colore uguale, il bianco è assegnato al giocatore con l'ARO (media Elo degli avversari incontrati) più alto.
- 1 punto per la vittoria, ½ per la patta, 0 per la sconfitta. Forfeit e BYE valgono un punto e la partita è considerata giocata senza avversario e colore (ai fini dello spareggio tecnico = patta contro se stesso)
- gli ID sono assegnati per Elo decrescente
- i ritardatari sono gestiti in modo dinamico, senza penalizzazioni
- se dispari, riposa l'ultimo giocatore della classifica parziale che non abbia mai avuto punti senza giocare
- a partire dalla testa della classifica parziale, si tenta di abbinare giocatori a pari punteggio. Se si forma un gruppo dispari, viene pescato un giocatore del gruppo successivo che bilanci i colori attesi, ma eccetto l'ultimo turno, lo stesso non può essere spostato due volte di fila o tre volte nel torneo, se max 10 turni, o quattro se più di 9 turni.

Se necessario all'interno di un gruppo di punteggio si pareggiano i colori attesi mediante forzature, ma nel rispetto delle regole sul colore ...

... quindi la prima metà (colore atteso bianco) viene ordinata per ARO crescente, Rating crescente e alfabeticamente; la seconda metà (colore atteso nero) viene ordinata per Rating decrescente, ARO decrescente e alfabeticamente. La prima metà viene abbinata con la seconda metà, eseguendo degli scambi nella seconda metà se la partita è già stata giocata.
Se risulta impossibile fare le coppie, si effettuano scambi nell'intero gruppo. Se ancora risulta impossibile fare le coppie, si aggiungono al gruppo di punteggio altri due giocatori del gruppo successivo; se si arriva alla fine della lista si inizia a spezzare le coppie già formate e si ripete il procedimento includendo tutti i giocatori non ancora appaiati.

n.b. L'implementazione è pienamente aderente alle regole della FIDE
(per dettagli si veda l'Handbook 2010 della FIDE presente nella cartella 'swiss')

Implementazione del motore di abbinamento 'Svizzero Semplice'

- il numero di turni è stabilito in anticipo. (può essere esteso)
- due giocatori possono incontrarsi una sola volta (può essere esteso)
- Le partite con il bianco e con il nero dovrebbero essere pari e alternate. Sequenze di tre partite con lo stesso colore o differenze maggiori di 2 sono consentite solo all'ultimo turno. In caso di storia di colore uguale, il giocatore con il più basso ID alterna l'ultimo colore giocato.
- 1 punto per la vittoria, ½ per la patta, 0 per la sconfitta. Forfeit e BYE valgono un punto e la partita è considerata giocata senza avversario e colore (ai fini dello spareggio tecnico = patta contro se stesso)
- gli ID sono assegnati per Elo decrescente
- i ritardatari sono gestiti in modo dinamico, senza penalizzazioni
- se dispari, riposa l'ultimo giocatore della classifica parziale che non abbia mai avuto punti senza giocare (*)
- a partire dalla testa della classifica parziale, si tenta di abbinare giocatori a pari punteggio. Se si forma un gruppo dispari, il giocatore compatibile (1. del colore dominante, 2. può avere un avversario valido, 3. possibilmente rispettando che ciò non avvenga due volte di fila o come due turni prima e che abbia il colore atteso opposto) viene spostato al gruppo successivo (*) ...

... quindi la prima metà viene abbinata con la seconda metà, eseguendo permutazioni nella seconda metà se ci sono vincoli di colore, cercando comunque di ottenere il maggior numero possibile di coppie con i colori attesi (*), o la partita è già stata giocata.
Se risulta impossibile fare le coppie, si effettuano scambi nell'intero gruppo. Se ancora risulta impossibile fare le coppie, si aggiungono al gruppo di punteggio altri due giocatori del gruppo successivo; se si arriva alla fine della lista si inizia a spezzare le coppie già formate e si ripete il procedimento includendo tutti i giocatori non ancora appaiati (*).

n.b. (*) = differenze implementative e semplificazioni introdotte rispetto allo 'Svizzero basato sul Rating'.

Implementazione del motore di abbinamento 'Svizzero a Perfetta Alternanza di Colore'

- il numero di turni è stabilito in anticipo. (può essere esteso)
- due giocatori possono incontrarsi una sola volta (può essere esteso)
- il primo e l'ultimo turno sono calcolati con le regole standard dello svizzero
- negli altri turni le partite con il bianco e con il nero sono sempre pari e alternate. Poiché i giocatori con ID dispari non si incontrano mai tra loro, come pure quelli con ID pari, bisogna ad un certo punto del torneo commutare su un sistema svizzero tradizionale (questo evita anche il formarsi di coppie troppo eterogenee per punteggio). Il programma avvisa quando mancano due turni alla fine oppure quando rimane al comando della classifica parziale un solo giocatore
- 1 punto per la vittoria, ½ per la patta, 0 per la sconfitta. Forfeit e BYE valgono un punto e la partita è considerata giocata senza avversario e colore (ai fini dello spareggio tecnico = patta contro se stesso)
- gli ID sono assegnati per Elo decrescente
- i ritardatari sono gestiti in modo dinamico, senza penalizzazioni
- se dispari, riposa l'ultimo giocatore della classifica parziale, del colore dominante, che non abbia mai avuto punti senza giocare
- Vengono creati due gruppi: S1 e S2 ciascuno contenente metà dei giocatori. S1 ha colore atteso bianco e viene ordinato per punteggio decrescente, ARO crescente e Rating crescente; S2 ha colore atteso nero e viene ordinato per punteggio decrescente, Rating decrescente e ARO decrescente. Infine S1 è abbinato a S2 con scambi in S2 se la partita è già stata giocata.

Implementazione in JavaPairing dell'algoritmo 'Amalfi Rating'

- il numero di turni è stabilito in anticipo. (può essere esteso)
- due giocatori possono incontrarsi una sola volta (può essere esteso)
- Le partite con il bianco e con il nero dovrebbero essere pari e alternate. Sequenze di tre partite con lo stesso colore o differenze maggiori di 2 sono consentite solo all'ultimo turno. In caso di storia di colore uguale, il giocatore con il più basso ID alterna l'ultimo colore giocato.
- 1 punto per la vittoria, ½ per la patta, 0 per la sconfitta. Forfeit e BYE valgono un punto e la partita è considerata giocata senza avversario e colore (ai fini dello spareggio tecnico = patta contro se stesso)
- gli ID sono assegnati per Elo decrescente
- i ritardatari sono gestiti in modo dinamico, senza penalizzazioni
- se dispari, riposa l'ultimo giocatore della classifica parziale che non abbia mai avuto punti senza giocare
- a partire dalla testa della classifica parziale, un giocatore è abbinato con quello che lo segue di un numero di posizioni uguale al numero di turni ancora da giocare. Se la coppia non è legale, a scalare viene scelto un giocatore più vicino o eventualmente più lontano. Se nessuna coppia è possibile, si rompe quella precedente e si ripete l'algoritmo.

n.b. Per una totale aderenza al regolamento ufficiale si prenda visione del documento: “Implementazione dell'algoritmo Amalfi Rating in JavaPairing. Note esplicative” presente nella cartella “Amalfi” della distribuzione

Questo e' un progetto aperto (licenza GNU/ GPL) e collaborativo.

Ogni contributo di programmazione, test, design grafico e traduzione dei testi è fortemente incoraggiato.

Ringrazio quanti mi hanno fin qui aiutato e sostenuto tramite contatti email e newsgroup e lo faranno ancora

Ringrazio quanti vorranno utilizzare il programma nella loro attività arbitrale

Ringrazio la CAF per l'invito di oggi e attendo fiducioso l'inserimento del programma tra i software autorizzati