Szachy i informatyka - praktyczne rozwiązania dla problemów na 64 polach

Stanisław Zawadzki

Wrocław, 31 maja 2011

Spis treści

1 Wstęp

1.1 Motywacja

W ciągu dwóch ostatnich dekad stworzonych zostało mnóstwo programów i witryn internetowych związanych z szachami. Realizują one różne funkcjonalność, implementują dziesiątki różnych interfejsów graficznych, umożliwiają niemal wszystkie czynności, jakie człowiek może wykonać z rozgrywkami na 64 polach. Większości z nich łączy jednak jedna wspólna cecha - skupiają się na zagadnieniu programistycznym, które w przypadku szachów jest często fascynujące, zapominając jednak o opcjach pozwalających użytkownikom na sprawne posługiwanie się ich narzędziami. Motywacją do mojej pracy było stworzenie wygodnego dla użytkowników serwisu internetowego pozwalającego na łączenie wielu funkcji związanych z szachami. Głównymi odbiorcami programu będą kibice oglądający przez Internet turnieje szachowe, ale ma być również użyteczny dla osób szukających informacji o turniejach szachowych, a także dla zainteresowanych profesjonalnymi analizami partii. Nie zamierzam zaprzestawać prac nad programem - w dalszej przyszłości chciałbym rozszerzyć spektrum zastosowań strony o możliwość bezpośredniej gry przez internet i kalendarz do wyszukiwania turniejów szachowych. Rozważam także ideę napisania nowej biblioteki do obsługi partii szachowych., dzięki której mógłbym realizować funkcje niedostępne w obecnych rozwiązaniach.

1.2 Słownik terminów szachowych

W trakcie tej pracy będzie przewijać się wiele terminów dotyczących szachów i programów szachowych. W tym rozdziale wyjaśnię znaczenie owych słów, a także podam listę wyrażeń do nich synonimicznych.

- **Szachy** - gra planszowa rozgrywana na szachownicach przy użyciu figur szachowych;
  Synonim: królewska gra;
- **Szachownica** - plansza, na której rozgrywa się partię szachów. Podzielona na 64 pola - 32 białe i 32 czarne. Zawsze posiada wymiary 8x8, przy czym kolumny są oznaczane literami alfabetu(a-h), podczas gdy rzędy są opisywane cyframi(1-8). Standardem opisywania pola jest podanie najpierw wartość kolumny, później zaś rzędu. Pole a1 jest zawsze czarnego koloru.
- **Figura szachowa** - Jedna z sześciu możliwych typów figur (pionek, goniec, skoczek, wieża, hetman, król). Zawodnicy przemieszczają je po szachownicy podczas swoich posunięć
  Synonim: bierka szachowa;figura
- **Posunięcie** - Zmiana położenia figury szachowej wykonana przy zachowaniu zasad poruszania się tych figur
  Synonim: ruch, półposunięcie
- **Szach** - Zaatakowanie króla przeciwnika, zmuszające go do natychmiastowej reakcji
  Synonim: atak na króla
- **Mat**[1] - Szach, po którym nie można nie uratować swojego króla. Kończy natychmiast partię jako wygrana zawodnika, który wykonał posunięcie matujące

[1]Wyrażenie „szach mat” wzięło się z języka staroperskiego - oznacza „król jest martwy”

- **Wygrana** - Aby wygrać partię należy zamatować przeciwnika lub skłonić go do dobrowolnego poddania partii. Zawodnik wygrywający otrzymuje jeden punkt, jego przeciwnik nie zyskuje żadnych punktów.

- **Remis** - Uznanie partii za nierozstrzygniętą. Skutkuje podziałem punktów między obu zawodników (obaj otrzymują wówczas $\frac{1}{2}$ punktu).
  Synonim: podział punktu

- **Propozycja remisu** - Dobrowolna oferta składana w dowolnym momencie gry przez jednego przeciwnika drugiemu. Jej zaakceptowanie oznacza podział punktu. Odrzucenie propozycji nie wpływa na przebieg rozgrywki

- **Reklamowanie remisu** - Istnieją sytuacje, w których można wnioskować o odgórne przyznanie remisu. Jeśli pozycja spełnia opisane w przepisach kryteria, wówczas sędzia powinien orzec podział punktu. W przeciwnym razie partii toczy się dalej.

- **Czas rozgrywki** - Choć nie jest to wymagane przepisami, w praktyce partie mają swój limit czasu. Partia trwa wówczas 2X minut, gdzie X oznacza czas dla jednego zawodnika. Czas zawodnikowi płynie wówczas gdy jest kolej na jego posunięcie. Do odmierzania czasu korzysta się z zegarów szachowych. W momencie jak zawodnikowi kończy się czas do namysłu przegrywa on partię w momencie zauważenia tego faktu przez przeciwnika.
  Synonim: tempo partii

- **Zegar szachowy** - Urządzenie składające się z dwóch czasomierzy i przełącznika między nimi. Początkowo czas na obu licznikach jest ten sam, jednak upływa on różnie u obu zawodników - w zależności od czasu spędzonego na namyśle nad swoimi posunięciami.

- **Czas dodawany** - W niektórych przypadkach wybierane jest tempo gry, które modyfikuje wartość czasu na zegarze po wykonaniu posunięcia. Jest to zazwyczaj wartość pomiędzy trzema a trzydziestoma sekundami. Dodaje się ona automatycznie po przełączeniu dźwigni zegara.

- **Zawodnik grający białymi** - Gracz rozpoczynający grę.
  Synonimy: Pierwszy zawodnik, gracz białymi, białe

- **Zawodnik grający czarnymi** - Gracz wykonujący posunięcia jako drugi.
  Synonimy: Drugi zawodnik, gracz czarnymi, czarne

- **Silnik szachowy** - Program oceniający pozycję i analizujący możliwości przebiegu partii.
  Synonimy: Program szachowy, silnik, silnik analityczny

- **Zapis partii** - Zapisanie ciągu wszystkich wykonanych posunięć wraz z opisem danych zawodników i zawodów

- **Szachownica elektroniczna** - Specjalna szachownica pozwalająca na automatyczne czytanie przebiegu partii

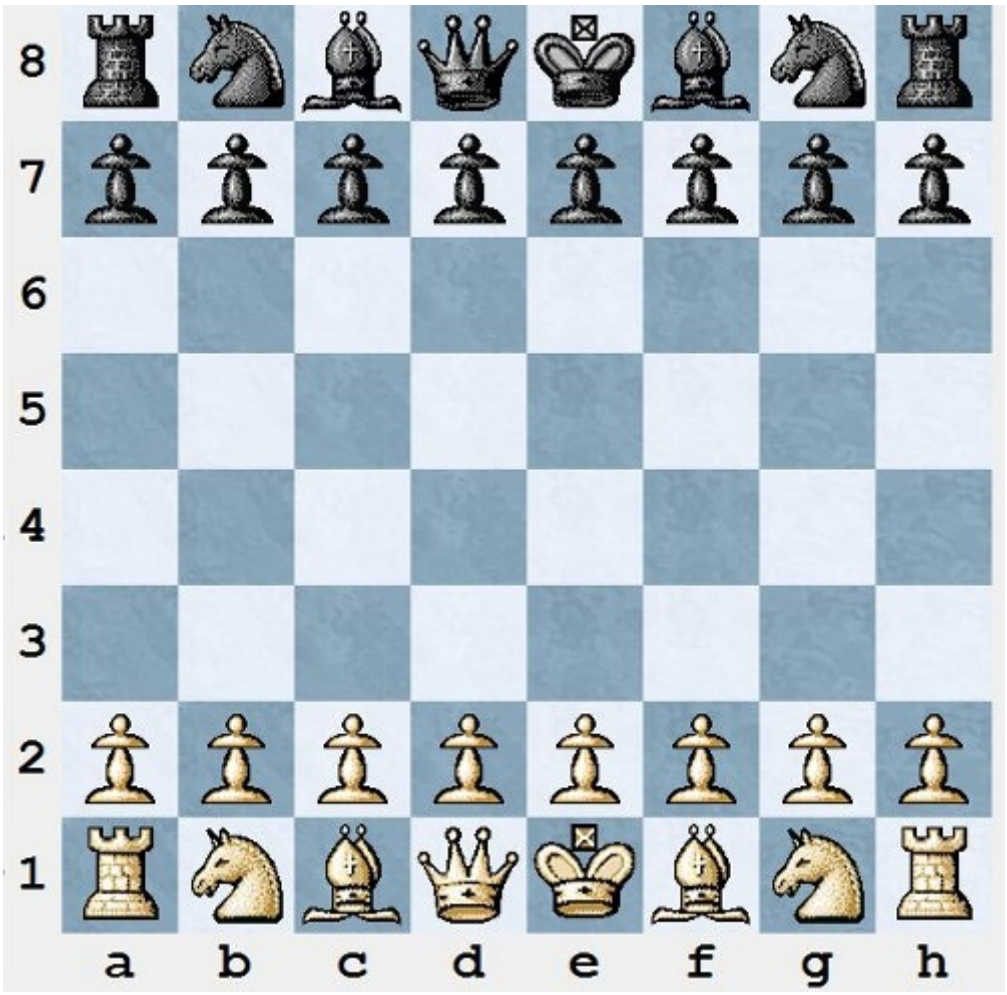
2 Terminologia szachowa

Aby móc rozmawiać o szachowych programach i interfejsach trzeba posiadać podstawową wiedzę na temat zasad królewskiej gry. W bieżącym rozdziale postaram się opisać każdą możliwą sytuację w partii szachowej, ale także opiszę specyfikę sprzętu i oprogramowania, które są wykorzystywane w praktyce turniejowej. Rozpocznę od opisu ruchu figur, później omówię posunięcia nadzwyczajne, następnie opiszę wszystkie inne aspekty, które są kluczowe w przypadku implementowania interfejsu szachowego. Na końcu zostaną opisane szachownice elektroniczne oraz silniki szachowe, które to umożliwią komputerowi znajdowanie potencjalnie najsilniejszych posunięć w pozycji.

2.1 Szachownica i figury

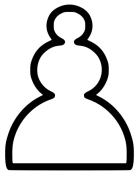
Choć zasady gry w szachy są wielu osobom znane, na potrzeby tej pracy chciałbym przytoczyć wszystkie informacje, które mają znaczenie dla interfejsów szachowych. Plansza, na której odbywa się rozgrywka jest nazywana szachownicą. Jest to obszar podzielony na 64 pola (8x8) numerowane w wierszach liczbami (1-8), w kolumnach zaś literami (a-h). W terminologii szachowej zarówno na wiersze, jak i na kolumny mówi się "linie". Patrząc z perspektywy gracza grającego białymi w lewym dolnym rogu jest pole a1, patrząc z perspektywy jego przeciwnika jest to pole h8. Pola są dwóch rodzajów - białe i czarne. Pola jednego koloru sąsiadują ze sobą po skosie i **pole a1 jest zawsze czarne**. Fizycznie nie ma wymagania, by używać dokładnie tych kolorów [2] i w przypadku programów szachowych standardem jest uznanie każdego koloru poza białym jako czarny.

[2] W rzeczywistości czarne pola są najczęściej ciemnobrązowe, białe zaś mają kolor jasnożółty.



2.2 Zasady poruszania się figur

2.2.1 Piony



W przypadku pionów sytuacja jest dość prosta. Te najliczniej reprezentowane figury mogą się przesuwać wyłącznie do przodu. Kiedy przypada kolej danego zawodnika może on wykonać pionkiem jedno z trzech możliwych posunięć:

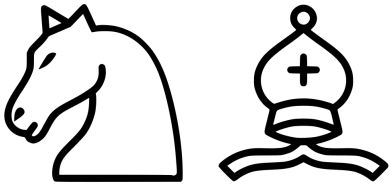
1. **Przesunąć pionka o jedno pole do przodu** (wzdłuż kolumny)
   Jest to podstawowe posunięcie możliwe do wykonania pionkiem. Można je wykonać wtedy, gdy pole na które zmierza pionek nie jest zajęte przez inną figurę (zarówno swoją, jak i cudzą).

2. **Przesunąć pionka o dwa pola do przodu**
   Posunięcie to można wykonać tylko wtedy, gdy pionek zajmuje początkową pozycję. Jak w poprzednim przykładzie pole, na które zmierza pionek, jak i pole pośrednie nie mogą być zajęte przez żadną figurę.

3. **Zbić figurę przeciwnika poruszając się po skosie do przodu**
   Pionki jako jedyne szachowe figury biją w inny sposób niż się poruszają. W zwyczajnej sytuacji pionki poruszają się wzdłuż kolumn. Jedynym sposobem, aby pionek przeszedł na sąsiednią linię jest zbicie innej figury. Pionki biją "po skosie", czyli jeśli pionek białych stoi na polu e3, to może zbić figurę, która stoi na d4 lub f4. W przypadku gdyby na e3 stał pionek czarnych, mógłby on zbić figurę na polu d2 lub f2. W wyniku zbicia stawiamy pionka na polu dotychczas zajmowanym przez daną figurę, tamtą zaś usuwamy z szachownicy

Dodatkowym i bardzo rzadkim ruchem jest tzw. „bicie w przelocie", o którym napiszę w dziale "Posunięcia niezwyczajne"

2.2.2 Lekkie figury

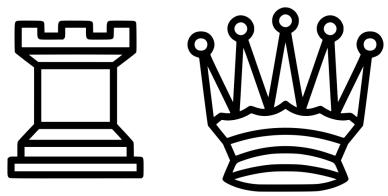


W nomenklaturze szachowej mianem „figury lekkiej" nazywane są gońce i skoczki. W tym punkcie zajmiemy się zdefiniowaniem sposobu poruszania się przez te dwie figury.

Goniec, zwany czasem błędnie laufrem, porusza się zawsze po diagonalach. Wynika z tego, że jeśli początkowo figura ta jest ustawiona na polu białego koloru, to do samego końca rozgrywki będzie się poruszała wyłącznie po polach tego właśnie typu(nazywamy go wówczas gońcem białopolowym). Goniec nie może „przeskoczyć nad przeszkodą". Stąd wynika, że może poruszyć się maksymalnie tak daleko, jak dużo ma wolnego miejsca po diagonali[3]. Goniec może zbić pierwszą figurę, która staje na jego drodze.

Skoczek jest jedyną figurą, która może przeskoczyć nad innymi figurami. Ruchy skoczka jest najłatwiej porównać do litery „L". Może on wykonać posunięcie w każdą stronę, pod warunkiem iż będzie to ruch opisany jako "dwa pola do przodu i jedno w bok". W przypadku skoczka zwrot „do przodu" oznacza dowolny z czterech kierunków. Bicie skoczkiem polega na tym, że skoczek może zabrać figurę z tego pola, na które uda mu się wskoczyć.

2.2.3 Ciężkie figury



Podobnie jak w przypadku lekkich figur termin „ciężkie figury" określa dwa rodzaje szachowych figur: wieże i hetmana. Ciężkie figury w zdecydowanej większości przypadków są silniejsze od pozostałych figur.

Posunięcia **wieżą** są podobne do posunięć gońca, z tą różnicą, że porusza się ona wzdłuż pionowych i poziomych linii. Tak jak w przypadku gońca może ona bić pierwszą figurę, która wystąpi na jej drodze. Innym ruchem, który może ona wykonać, jest roszada opisana w dziale

[3]W nomenklaturze szachowej każda ukośna linia mająca przynajmniej trzy pola jest diagonalą

posunięć niezwyczajnych.

Posunięcia **hemtanem** są równie proste jak posunięcia wieżą lub gońcem - łączy on cechy obu tych figur, a do tego nie posiada żadnych posunięć niezwyczajnych.

2.2.4 Król

Król jest, w zależności od punktu widzenia, albo najsłabszą, albo najsilniejszą figurą. Porusza się on w bardzo prosty sposób - może przejść na każde z okolicznych pól. Jedynym dodatkowym posunięciem w jego arsenale jest wzmiankowana wcześniej i opisana w następnym podrozdziale roszada. Jest jednak bardzo ważna cecha, która odróżnia króla od każdej innej figury. Monarcha nie może zostać umieszczony na polu atakowanym przez chociaż jedną figurę rywala. Tak samo trzeba zawsze coś przedsięwziąć, jeśli przeciwnik wykonując ruch zaatakował naszego króla. W tym przypadku **trzeba** wykonać jeden z trzech manewrów

- odejść królem na nieatakowane pole;
- zbić atakującą figurę;
- przesłonić obszar działania atakującej figury.

Król jest jedyną figurą z naszego arsenału, której nie możemy stracić. W momencie gdy monarcha zostaje tak osaczony, że nie może uniknąć zagłady, następuje koniec partii. Więcej o zakończeniu partii szachowej będzie napisane w następnych rozdziałach.

2.3 Posunięcia niezwyczajne

W poprzednim punkcie przedstawiłem wszystkie zwyczajne sposoby na wykonanie posunięcia. Teraz nadszedł czas na opisanie przypadków posunięć nadzwyczajnych, posunięć, których wykonanie często zależne jest od wcześniejszego przebiegu rozgrywki. Niektóre z nich zdarzają się bardzo rzadko, niektóre z nich są obecne w prawie każdej partii szachowej.

2.3.1 Bicie w przelocie

Bicie w przelocie (ang. *en passant*) jest najrzadszym szachowym posunięciem. Może ono wystąpić tylko w jednym przypadku. Warunkiem koniecznym i wystarczającym, aby można było zagrać ten ruch, jest fakt, iż pionek przeciwnika ruszył się o dwa pola do przodu, zaś nasz pionek atakuje pośrednie pole między początkowym o końcowym ustawieniem pionka rywala. Aby dokonać zbicia przesuwamy pionka na owo pole pośrednie i zdejmujemy pionka rywala (jest to jedyny przypadek w grze, że bije się figurę z pola, na którym nie postawiło się swojej figury). Ważną sprawą jest to, że ruch ten można wykonać tylko bezpośrednio po przesunięciu pionka o dwa pola. Później takie bicie jest niezgodne z przepisami.

2.3.2 Roszada

Roszada, w przeciwieństwie do bicia w przelocie, choć jest posunięciem niezwyczajnym, zdarza się bardzo często. Jest ona jedynym szachowym posunięciem, które wprawie w ruch więcej niż jedną figurę. Polega ona na jednoczesnym przesunięciu króla i wieży po pierwsze linii (odpowiednio dla czarnych figur - ostatniej). Muszą być jednak spełnione następujące warunki:

1. Wszystkie pola między królem a wieżą muszą być puste;
2. Zarówno król, jak i wieża nie mogły wcześniej wykonać ani jednego posunięcia;
3. Król nie jest atakowany przed wykonaniem roszady, nie będzie zaatakowany po wykonaniu roszady i nie przekroczy zaatakowanego pola w trakcie jej wykonywania.

Oczywiście drugi z tych punktów determinuje, że roszadę jeden zawodnik może wykonać maksymalnie raz w czasie całej partii.[4]
Ponieważ są dwie wieże na szachownicy to rozróżniamy dwa typy roszad.

- roszada krótka (występująca częściej w partiach szachowych) - król przechodzi z pola e1 na pole g1, wieża przesuwa się z h1 na f1
- roszada długa - król przechodzi z pola e1 na pole c1, wieża przesuwa się z pola a1 na pole d1 (zauważmy, że w przypadku tej roszady pola b1/b8 **może** być atakowane).

2.3.3 Promocja

Promocji dokonuje się z pomocą przesunięcia pionka aż do końcowej linii szachownicy (ósmej w przypadku białego pionka, pierwszej w przypadku czarnego). W momencie gdy pionek osiąga skraj szachownicy **musi** on zostać zmieniony na hetmana, wieżę, gońca lub skoczka **tego samego koloru!** Dozwolone jest posiadanie więcej figur, niż określa to stan początkowy (czyli można mieć na szachownicy nawet dziewięć hetmanów). Promocji można dokonać każdym pionkiem i cała czynność, czyli postawienie pionka na ostatniej linii oraz zamienienie go na figurę trwa dokładnie jedno posunięcie.

2.4 Początek, przebieg i koniec rozgrywki

Nadszedł czas na opisanie warunków początkowych i końcowych rozgrywki szachowej. Zwłaszcza zakończenie partii jest ważną kwestią, gdyż istnieje duża liczba sytuacji, w których partię uznaje się za skończoną.

2.4.1 Rozpoczęcie rozgrywki

Na początku figury są rozstawione w sposób przedstawiony na diagramie nr XXX. Po zwyczajowym przywitaniu się z przeciwnikiem następuje uruchomienie zegara szachowego i białe wykonują swój pierwszy ruch. Od tej chwili obaj zawodnicy wykonują na zmianę po jednym posunięciu, korzystając z opisanych we wcześniejszym rozdziale zasad. W momencie gdy zawodnik zaatakuje króla rywala, powinien powiedzieć "szach", aby ów wiedział, że ma obowiązek zaradzeniu temu problemowi. Posunięcia są wykonywane na zmianę aż do momentu,

[4]Choć w praktyce turniejowej zdarzały się przepadki, iż zawodnicy w stresie zapomnieli o tym, że roszada miała lub miejsce lub król wykonał już posunięcie i w wyniku tego partia zaczęła się toczyć w sposób nieprawidłowy

aż w jakiś sposób nastąpi koniec partii - w przypadku gdy w pewnym momencie nastąpił nieprawidłowy ruch powraca się do ostatniej prawidłowej pozycji. Sposobów, aby zakończyć partię jest naprawdę wiele, każda z nich jest szczegółowo opisana w przepisach Międzynarodowej Federacji Szachowej (FIDE).

2.4.2 Różne sposoby zakończenia partii

Partia standardowo może zakończyć się na dwa sposoby - remisem lub zwycięstwem jednej ze stron. W szczególnych przypadkach dozwolone są jednak inne wyniki, takie jak walkower (oznaczany + - jeśli walkowera oddały czarne i w -+ w przeciwnym przypadku), obustronna porażka (oznaczana 0-0, przyznawana w przypadku skrajnego złamania zasad fair-play) oraz wynik $\frac{1}{2}$ - 0 (lub 0 - $\frac{1}{2}$) , który może się zdarzyć w wyjątkowych sytuacjach (jednemu z zawodników pozostał tylko król, drugi wciąż posiada materiał matujący, ale przekracza czas przeznaczony do namysłu). Trzeba jednak pamiętać, że wyniki niestandardowe zdarzają się w znikomym ułamku liczby partii (najczęściej z nich widuje się walkowera) Zakończenie partii określają przepisy FIDE:

ARTYKUŁ 5. PARTIA ZAKOŃCZONA

[5] **5.1. (a)** Partię wygrywa zawodnik, który zamatował króla przeciwnika. Mat natychmiast kończy partię pod warunkiem, że pozycja matowa powstała w wyniku prawidłowego posunięcia.

(b) Partię wygrywa zawodnik, którego przeciwnik oświadczył, że poddaje się. Poddanie natychmiast kończy partię. **5.2 (a)** Partia kończy się remisem, gdy zawodnik będący na posunięciu nie może wykonać żadnego prawidłowego ruchu, a jego król nie jest szachowany. Taką pozycję określa się słowem „pat". Pat natychmiast kończy partię, pod warunkiem, że pozycja patowa powstała w wyniku prawidłowego posunięcia.

(b) Partia kończy się remisem, jeśli powstaje pozycja, w której żaden z zawodników nie może zamatować króla za pomocą jakiejkolwiek serii prawidłowych posunięć. Pojawienie się tzw. „martwej pozycji" też natychmiast kończy partię, pod warunkiem, że powstała ona w wyniku prawidłowego posunięcia. (Patrz art. 9.6)

(c) Partia kończy się remisem w wyniku uzgodnienia pomiędzy obydwoma zawodnikami w trakcie partii. Zgoda na remis natychmiast kończy partię. (Patrz art. 9.1).

(d) Partia może zakończyć się remisem, jeżeli identyczna pozycja ma pojawić się na szachownicy przynajmniej po raz trzeci lub już pojawiła się przynajmniej trzykrotnie. (Patrz art. 9.2).

(e) Partia może zakończyć się remisem, jeżeli obaj zawodnicy przynajmniej w ostatnich 50 posunięciach nie wykonali żadnego ruchu pionkiem, ani nie pobili żadnej bierki. (Patrz art. 9.3).

Trzeba zawsze pamiętać o każdym z wymienionych sposobów i posiadać narzędzia pozwalające sprawdzić, czy strony mogą zgodzić się na podział punktu. W ostatnich latach spora część zawodów szachowych dodała zasadę zakazu proponowania remisu przed zagraniem 30 posunięć. Jest to jednak wciąż jedynie wewnętrzny przepis i organizatorzy turnieju nie mają obowiązku się do niego stosować - oficjalnie wystarczy, że zawodnicy zagrają dwa półruchy[6]

[5] Przepisy gry od 01.07.2009 wg Światowej Federacji Szachów

[6] Wcześniej wg definicji partia nie jest jeszcze uznana za rozpoczętą

Nie licząc sytuacji zupełnie losowych[7] oraz wyjątkowych, określanych w osobnych przepisach partie mogą kończyć się wyłącznie w sposoby określone w wyżej przytoczonych przepisach.

2.5 Dodatkowe informacje o partiach szachowych

Pisząc o partiach szachowych należy przytoczyć jeszcze kilka terminów, które dotyczą niemal każdej rozgrywki. Nie mają one bezpośredniego związku z szachami i można towarzysko rozgrywać partię bez tych elementów, jednak w profesjonalnych turniejach są one najczęściej wymagane przepisami.

2.5.1 Czas rozgrywki

W zdecydowanej większości przypadków partie szachowe, nawet te towarzyskie, grane są z limitem czasu. Może to być zaledwie minuta dla zawodnika na partię, a może to być kilka godzin. W ostatnich latach coraz bardziej popularna jest opcja dodawania czasu po wykonaniu każdego posunięcia. Zawodnicy mogą zużyć swój czas w zależności od potrzeb na analizowanie swoich posunięć. W momencie gdy wykonają ruch przełączają upływ czasu na konto rywala. W przypadku, jeśli zabraknie czasu do namysłu, partia jest uznana za przegraną zawodnika, który przekroczył limit. Historycznie czas odmierzano na zegarach mechanicznych, jednak coraz bardziej sie od tej praktyki odchodzi na rzecz elektronicznego mierzenia upływu czasu, pokazującego stan zegara z dokładnością do sekundy.

2.5.2 Zapis rozgrywki

W partiach turniejowych granych dłuższym tempem (minimum godzina dla jednego zawodnika) istnieje obowiązek prowadzenia zapisu partii. Służy on zapamiętywaniu stanu partii po każdym wykonanym posunięciu, aby w każdej chwili można było prześledzić historię rozgrywki. Ogólnoświatową konwencją jest stosowanie tzw. skróconej notacji algebraicznej. Na początku zapisywana jest pierwsza litera figury[8], później zaś pole, na które ona zmierza. Jeśli jest to nierozstrzygające dopisuje się jeszcze informację, która z figur[9] W świecie rzeczywistym odbywa się to z użyciem papierowych blankietów, w świecie programów komputerowych przechowuje się wiadomości o partii z użyciem plików o określonej strukturze.

2.6 PGN i FEN - sposoby zapamiętywania partii

Od kiedy tylko informatyka włączyła się komercyjnie w świat szachów powstało pytanie, jak skutecznie zapisywać i odczytywać pojedyncze rozgrywki szachowe. W tym celu powstał format **PGN (Portable Game Notation)**[10], w którym w prosty sposób można zapisać wszystkie kluczowe informacje o partii szachowej.
PGN jest formatem tekstowym, czytelnym dla bardziej doświadczonego szachisty. Nawet nie dysponując żadnym programem, który graficznie przedstawiłby daną rozgrywkę, bez najmniejszych problemów można otworzyć ten plik w edytorze tekstowym i przeczytać nagłówek i posunięcia. Format ten sprawdza się dobrze w przypadku pojedynczych partii lub zbiorów nie przekraczających kilkadziesięciu tysięcy rozgrywek, jednak w przypadku wielkich

[7] Zdarzały się nawet przypadki śmierci przy szachownicy

[8] W niektórych językach, jak n.p. w angielskim jest to nie zawsze możliwe. Wówczas stosuje się inne oznaczenie

[9] Na przykład Sde2 - skoczek z linii 'd' idzie na pole e2

[10] dokumentację można znaleźć m.in. pod adresem http://pgn.freeservers.com/Standard.txt

baz (kilka milionów partii) staje się bardzo nieefektywny z uwagi na zerową kompresję i brak wsparcia dla algorytmów wyszukiwania.[11] Plik PGN ma bardzo prostą strukturę - składa się z zapisów partii oddzielonych znakami nowej linii. Zapis każdej partii składa się z dwóch części: nagłówka i treści partii. W nagłówku podawane są informacje na temat warunków grania partii(miejsce, data, turniej czy runda turnieju), zaś w treści partii wypisane są posunięcia a także możliwe warianty i komentarze słowne. Dane w nagłówku przedstawione są w postaci [klucz "wartość"], przy czym obecność części kluczy nie jest wymagana. Posunięcia i warianty podawane są w skróconej lub rozszerzonej notacji algebraicznej. W samych plikach nie ma żadnej kontroli poprawności danych, zadanie to jest przerzucone na programistów korzystających z owych plików.

FEN (Forsyth-Edwards Notation) służy nie do zapisu partii, ale do zapamiętania konkretnej sytuacji na szachownicy. Czasem bowiem mamy do czynienia z sytuacją, kiedy nie potrzebujemy wcześniejszego przebiegu rozgrywki i interesuje nas tylko konkretna pozycja. Początkowo wydaje się to trywialnym zadaniem, jednak po głębszym namyśle odkrywamy problemy - jak zapisać, czy któraś ze stron może jeszcze zrobić roszadę (nie wynika to z samego ustawienia, bo król mógł używając przynajmniej dwóch posunięć raz jeszcze znaleźć się na pozycji początkowej), czy też możliwość bicia w przelocie? Do tego trzeba zadbać o zwięzłość tego zapisu. Notacja FEN opisuje linia po linii, każde pole na szachownicy, zliczając pola puste.

Przykładowo - pozycja z pięcioma figurami, białym królem na polu g5, hetmanem na polu e6 oraz czarnymi figurami na polach g7(król), f7 (wieża) i h7 (pionek) wygląda następująco "8/5rkp/4Q3/6K1/8/8/8/8 w - - 0 1"

Znak „/" oznacza nową linię poziomą szachownicy, wielkie litery opisują białe figury, małe litery opisują figury czarne. Litera 'w' oznacza, że grę rozpoczynają białe. Myślniki świadczą o braku możliwości zrobienia roszady.

W ten sposób, korzystając z zaledwie kilkunastu znaków opisaliśmy całą pozycję. W przypadku większej liczby figur na szachownicy czasem zapis notacji jest nieco dłuższy, ale nigdy nie jest to więcej jak kilkadziesiąt znaków. Z notacji FEN często korzysta się w plikach PGN, gdyż dzięki niej można rozpocząć zapis partii od dowolnej pozycji, a nie tylko od początkowej.

2.7 Oznaczenia figur szachowych

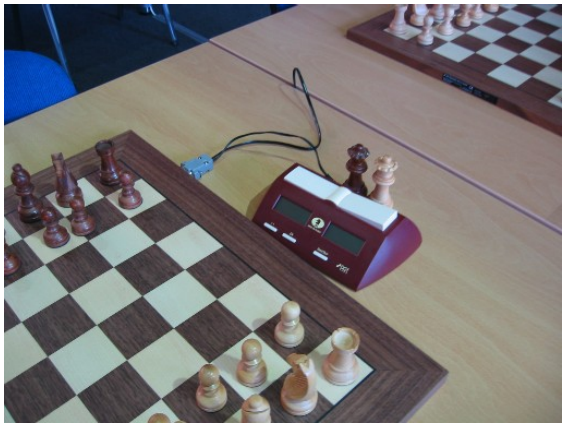
W formacie PGN należy rozstrzygnąć w jaki sposób oznaczamy figury W szachach nie ma jednolitego, międzynarodowego nazewnictwa. Każdy kraj stosuje nazwy w swoim języku. Jedynym standardem jest omijanie przedrostka, w przypadku gdy dochodzi do posunięcia pionkiem. Każdy profesjonalny szachista potrafi korzystać z nazw figur w języku angielskim[12]: król-K(king),hetman-Q(queen),wieża-R(rook),goniec-B(bishop), skoczek-N(knight). Nie należy jednak zakładać, że każdy użytkownik będzie posiadał tę wiedzę. Ciekawym rozwiązaniem jest korzystanie z piktogramów, które nie tylko już są pewnym standardem[13], ale są także graficznie zbliżone do rzeczywistych figur.

[11] Wówczas stosowane są inne, komercyjne formaty.

[12] Nie należy jednak tego mylić z tzw. notacją angielską, która jest obecnie już nieużywanym sposobem zapisywania posunięć

[13] Istnieją choćby w domyślnych czcionkach systemów biurowych jak Microsoft Word czy OpenOffice

2.8 Szachownice elektroniczne



Jeśli myślimy o transmisjach partii szachowych na żywo musimy także zrozumieć sposób, w jaki posunięcia szachowe są wysyłane do sieci. W tym celu korzysta się ze specjalnego sprzętu zwanego szachownicami elektronicznymi[14]. Z wyglądu przypominają one zwykłe szachownice, jednak podłączone są one przewodowo[15] do komputera, który przy pomocy programu autorstwa firmy DGT odczytuje sygnały z szachownic i interpretuje je jako posunięcia. Wówczas można już te partie zapisywać do plików i wysyłać na serwery do odczytu przez programy prowadzące transmisje.

Podczas prowadzenia transmisji z użyciem szachownic elektronicznych zdarzają się błędy, które można podzielić na kilka kategorii:

- Błędy fizyczne
  Na salach turniejowych (wyłączywszy nieliczne superturnieje) nie ma możliwości, aby kable łączące szachownice elektroniczne z komputerem poprowadzić pod podłogą lub w inny sposób uniemożliwiający przypadkowe ich zerwanie. Z tego powodu trzeba brać pod uwagę, iż któryś z kibiców lub zawodników[16] spowoduje zerwanie kabla z transmisji. Niesprawne lub niepoprawnie przekazujące dane zegary także są częstą przyczyną błędów w przypadku oglądania partii on-line;

- Błędy szachistów
  Doświadczeni szachiści bardzo rzadko popełniają błąd polegający na wykonaniu nieprawidłowego posunięcia[17]. Mimo to, zwłaszcza w przypadku ograniczonego czasu do namysłu, zdarzają się sytuacje, w której zawodnik wykona nielegalne posunięcie[18]. Spora część programów transmitujących partie nie bierze pod uwagę takich sytuacji. Powoduje to, że dalsza część partii jest odrzucana przez aplikację jako nieprawidłowa. Skutkuje to nie tylko zawieszeniem transmisji, ale także nie przekazuje żadnego komunikatu widzom, dlaczego stan szachownicy się nie zmienia;

- Błędy sędziego/obsługi technicznej zawodów
  Kolejną osobą, która może popełnić błąd, jest sędzia lub specjalista od obsługi technicz-

[14] Aktualnie rynek szachownic elektronicznych jest zmonopolizowany przez holenderską firmę DGT znaną także z produkcji najlepszych zegarów elektronicznych

[15] Przez gniazdo COM

[16] W czasie namysłu rywala wielu szachistów wstaje od swojej partii i spaceruje po sali gry

[17] Amatorzy dla odmiany rzadko grają na szachownicach elektronicznych, gdyż jest to drogi sprzęt i z tego powodu jest używany niemal wyłącznie na prestiżowych zawodach

[18] Najczęściej jest to wykonanie nieprawidłowej roszady, choć zdarzają się też inne możliwości jak przypadkowe wykonanie ruchu złą figurą

nej zawodów. Mając dostęp do panelu administrowania szachownicami może wpisać źle dane partii, może także wpisać zły wynik. Może oczywiście również źle skonfigurować sieć, przez co pliki nie będą prawidłowo umieszczane na serwerze.

- Błąd „późniejszej analizy”
  Jest to bardzo częsty błąd, który musi być starannie usuwany przez sędziego. Bierze się on z zachowania szachistów, którzy po zakończonej partii często przesuwają figury, aby sprawdzić potencjalny rozwój sytuacji. Komputer cały czas zapisuje zmiany na szachownicy jeśli nie otrzyma informacji, że obaj zawodnicy uzgodnili już słownie wynik partii. Co prawda teoretycznie unika się tego problemu poprzez specyficzne ustawienie królów zaraz po zakończeniu partii[19], ale w ferworze walki szachiści bardzo często o tym zapominają. Analizy po zakończeniu partii są bardzo często algorytmicznie nierozróżnialne od samej partii, zatem sędzia musi ręcznie, na podstawie papierowych zapisów partii, zweryfikować jej zakończenie. Kolejny problem tego samego rodzaju wynika, jeśli w końcowej pozycji postawienie króla na jednym z wyróżnionych pól jest możliwe. Wówczas ustawienie króla w celu wysłania informacji o wyniku zostanie potraktowane jako zwyczajne posunięcie królem.

2.9 Silniki szachowe

Silniki szachowe są programami, które jako dane wejściowe otrzymują pozycję, jako rezultat zwracają zaś ocenę pozycji wraz z propozycjami najlepszych posunięć. Nie należy jednak podchodzić do nich w sposób jaki podchodzi się do zwyczajnych programów - ich wynik w zasadzie nigdy nie jest rozstrzygający i pewny, a ich działanie może być pod względem praktycznym nieskończone. Bierze się to z olbrzymiej ilości możliwych kontynuacji z danej pozycji. Mimo iż szachownica posiada zaledwie 64 pola, a każda z figur posada ograniczone pole działania przestrzeń możliwych sekwencji ruchów jest olbrzymia. Zacznijmy jednak od udowodnienia prostego faktu.

[19]Biały król na e4, czarny na d5 w przypadku wygranej białych, biały król na e4, czarny król na e5 w przypadku remisu oraz biały król na d4, czarny na e5 w przypadku wygranej czarnych

Liczba wszystkich możliwych partii szachowych jest skończona. Wynika to jednoznacznie z trzech obserwacji:

1. Ilość figur na szachownicy może się jedynie zwiększać [20]
2. Każdy pionek ma skończoną możliwą liczbę posunięć - może wykonać maksymalnie sześć ruchów
3. W przypadku zagrania 50 posunięć bez bicia lub posunięcia pionkiem jest ogłaszany remis.

Z punktów 1. i 2. wynika, iż ilość bić i posunięć pionkiem jest ograniczona do 30(zbić)+ $(2 * 8 * 6)$(możliwe posunięcia wszystkich pionków), czyli łącznie możemy mieć maksymalnie 126 podobnego typu posunięć, co w połączeniu z punktem trzecim daje nam maksymalnie $50 * 126 = 6300$ posunięć w jednej partii szachowej. Udowadnia to, że szachy są grą skończoną.[21]

Mimo udowodnienia faktu, że każda partia szachowa musi się kiedyś zakończyć, trzeba być świadomym jak olbrzymia przestrzeń możliwych kontynuacji. Shannon szacuje się, że możliwych pozycji jest około 2^{43}[2], zaś Tromp udowodnił górną granicę na liczbę możliwych partii na 2^{155}[3]. Jest to oczywiście poza zasięgiem zdolności obliczeniowej komputerów. Nawet jeśli podejdzie się do sprawy bardziej praktycznie, eliminując za pomocą heurystyk jawnie błędne kontynuacje, to ilość możliwości jest wyzwaniem nawet dla współczesnych procesorów. Mimo iż komputery w chwili obecnej bez problemu pokonują najlepszych szachistów świata[22], to nie należy zwieść się iluzji o ich wszechpotędze. Programy osiągają przewagę nad ludźmi dlatego, że nie popełniają błędów i cały czas grają na równym, bardzo wysokim poziomie.[23] Cały czas jednak istnieją wyzwania, z którymi poradził sobie umysł ludzki, a maszyna wciąż nie potrafi.

Historia silników szachowych sięga lat 50-tych ubiegłego wieku, jednak dopiero w latach 80-tych maszyny zaczęły prezentować poziom, który pozwalał im na nawiązanie walki z profesjonalnymi szachistami. Lata dziewięćdziesiąte były czasem walki o supremacje między ludźmi i maszynami, zaś pierwsza dekada obecnego stulecia doprowadziła do sytuacji, że człowiek jest już bez szans w pojedynku z maszyną. Prawda jest jednak taka, że podstawowe zasady programowania silników szachowych pozostały podobne do tych, które rozwijali uczeni kilkadziesiąt lat temu.

Istnieją dwa podstawowe typy filozofii tworzenia silników szachowych. Pierwsze działają korzystając z algorytmów *brute-force*, drugie zaś dokonują wstępnej selekcji możliwych posunięć. Co może dziwić, przez wiele lat programy z pierwszej z tych grup osiągały lepsze wyniki. Dlaczego tak się działo? Odpowiedź jest prosta - bardzo trudno dokonać poprawnej selekcji posunięć. Nawet jeśli w 99% przypadków program dokona słusznej eliminacji, wciąż w wielu momentach partii nie będzie on rozpatrywał najsilniejszego posunięcia[24]. W takim

[20]Promocja może zwiększyć jakość figur, jednak nigdy nie zwiększy ich ilości

[21]Wyliczenie nie mają wiele wspólnego z praktyką. Rekord najdłuższej partii świata wynosi 279 posunięć, rekord Polski ustanowiony w partii Zawadzka-Lach, Warszawa 2011 wynosi 212 posunięć

[22]Dekadę temu Garri Kasparow toczył ciężkie boje z superkomputerem Deep Blue, w chwili obecnej pokonałby go zapewne komercyjny program uruchomiony na zwyczajnym komputerze osobistym

[23]Szachy są grą błędów. Najlepsi szachiści świata także popełniają wiele błędów - mniej jednak niż ich konkurenci

[24]Selekcja nie tylko odbywa się w korzeniu obliczeń, ale w większości jego potomnych węzłów również

przypadku zysk z eliminacji, czyli możliwość spojrzenia dalej w przyszłość, jest niwelowany przez ominięcie kluczowych wariantów. Drugi sposób działa w sposób przypominający ludzkie myślenie, jednak przez lata doprowadzało to do sytuacji, że programy korzystające z heurystyk selekcji posunięć były mniej konsekwentne niż programy pierwszego rodzaju, zaś wciąż zbyt słabo naśladujące umysł profesjonalnego szachisty. Jednak koniec końców programistom udało się doprowadzić do sukcesu drugiej grupy programów.

2.10 Algorytmy stosowane w silnikach szachowych

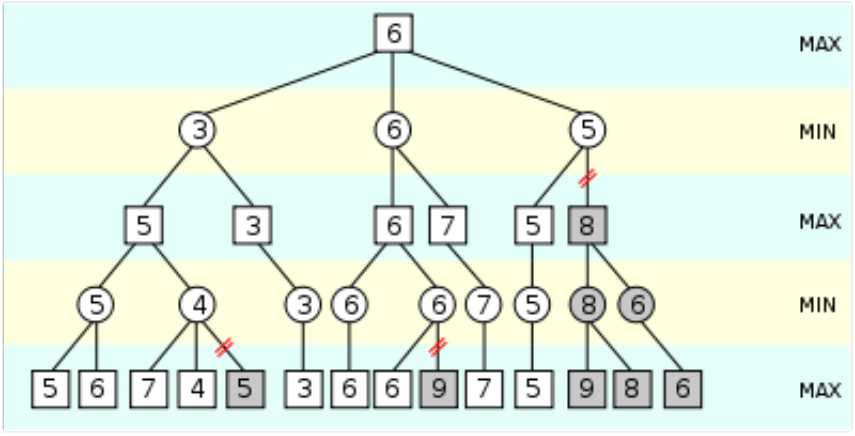
Silniki szachowe przedstawiają partię szachową jako drzewo. Cytując profesora Andrzeja Kisielewicza[4] *"Początkowym wierzchołkiem drzewa (korzeniem) jest stan początkowy gry (wraz ze wskazaniem , na kogo przypada ruch; przypomnijmy, że gracz zaczynający nazywa się MAX). Następnikami korzenia są wszystkie stany, jakie mogą powstać po wykonaniu ruchu przez MAX-a (w szachach jest ich 20). W każdym z tych stanów ruch przypada z kolei na MIN-a i następniki wyznaczone są przez wszystkie możliwe ruchy MIN-a. Proces ten kontynuujemy, wyznaczając dla każdego stanu następniki, którymi są wszystkie stany, jakie mogą powstać z danego stanu po wykonaniu ruchu przez gracza, na którego przypada kolej ruchu".*

Obliczenia działają w sposób typowy dla algorytmów MIN-MAX-owych. Na początku obliczane są wartości w liściach (w przypadku szachów 1,0,-1 jeśli zakładamy, że możemy przedstawić pełne drzewo szachów) Zawodnik o nazwie MAX stara się maksymalizować wartość w wierzchołku, podczas gdy MIN dąży do minimalizacji tej wartości. W ten sposób przesuwamy sie od liści aż do korzenia, który jest końcową wartością działania algorytmu. Jest to jednak teoretyczny przykład, gdyż dla zdecydowanej większości przypadków nie jesteśmy w stanie stworzyć tak wielkiego drzewa.
"Ta sytuacja nie zraziła badaczy AI (sztuczna inteligencja przyp. aut.)*, a raczej zdopingowała, pokazując, że tu jest właśnie miejsce na wprowadzenie sztucznej inteligencji. Komputer(program) powinien być zdolny do oceny aktualnej sytuacji, do stwierdzenia czy przewagę (lepsze widoki na zwycięstwo) mają białe, czy czarne. Następnie mógłby zanalizować różne sytuacje kilka lub kilkanaście ruchów naprzód i wybrać najbardziej obiecującą"* [4]

Silnik szachowe działają właśnie na zasadzie podobnej opisanej w powyższym akapicie. Twórcy rezygnują z nierealnego planu stworzenia pełnego drzewa szachów, budują jednak strukturę na kilkanaście półruchów naprzód i dając użytkownikowi częściowe wyniki obliczane za pomocą algorytmu MIN-MAX, gdzie wartości liścia wylicza się specjalnymi funkcjami ewaluacyjnymi. Funkcje te uruchamiane są dla każdej pozycji końcowej i z użyciem różnych kryteriów (tworzonych bardzo często przez zawodowych szachistów) liniowo wyliczają wartość pozycji. Jest to wartość orientacyjna, gdyż stworzenie dobrej funkcji ewaluacyjnej jest wyjątkowo trudnym zadaniem. Warto zwrócić uwagę, że algorytm tworząc liść od korzenia od razu wylicza wartość funkcji dla późniejszych węzłów, gdyż każdy wierzchołek kiedyś był liściem.

Aby uzyskać dobre praktyczne wyniki trzeba jeszcze bardziej usprawnić działanie algorytmu. W sytuacji przedstawionej wyżej komputer jest w stanie doliczyć maksymalnie do kilkunastu posunięć wprzód. Są pozycje, w której to jest zbyt mała wartość. W tym przypadku może pomóc metoda nazwana "odcinaniem alfa-beta" *(alpha-beta pruning)*. Wykorzystuje ona fakt, że jesteśmy w stanie dynamicznie podczas tworzenia drzewa wykazać, że dana gałąź nie rokuje szans na wygenerowanie końcowego rozwiązania. Wówczas algorytm MIN-MAX może uciąć tę gałąź, zmniejszając liczbę węzłów do przeszukania. Jest to poważne usprawnienie standardowego algorytmu MIN-MAX, tym bardziej, że w pozycji szachowej

bardzo często jest wiele ruchów, które można niemal automatycznie odrzucić. Dzięki odcinaniu możemy nie interesować się nie rokującymi nadziei na wynik gałęziami i wykorzystać zasoby do przeliczania bardziej szansownych kontynuacji.



W ostatnich latach silniki szachowe zostały wzbogacone o dodatkowe elementy. Z jednej strony dodano do nich programy zawierające wiedzę dotycząca debiutów i końcówek, z drugiej w samych funkcjach oceniających dodano podprogramy. Służą one do pokonywania słabości algorytmu MIN-MAX - problemu dalekiego horyzontu oraz rozpoznawania twierdz. Problem horyzontu następuje wówczas, jak pozycja ma jasną i stabilną ocenę, jednak powód tej oceny uwidoczni się dopiero za kilkanaście ruchów. Problem twierdz był natomiast zgoła odwrotny - komputer nie potrafił zauważyć, że pomimo bycia stroną silniejszą nie potrafi on wzmocnić pozycji tak, żeby osiągnąć zwycięstwo. Doprowadzało to do kuriozalnych sytuacji, kiedy program nakazywał chodzić w kółko nie mogąc przeprowadzić konstruktywnego planu, lecz cały czas uznawał swoją pozycję za wygraną. Obecne programy szachowe już radzą sobie z tym zadaniem.

3 Ewaluacja dotychczasowych rozwiązań

Programy służące do przeglądania partii szachowych istnieją już od dziesięcioleci. Profesjonalni szachiści wraz ze sporą liczba amatorów korzystają z całej gamy obecnych na rynku produktów. Niemal każdy szachista, który wyszukuje w Internecie informacje o szachach korzysta także z programów realizujących inne funkcje.

3.1 Kafejki szachowe

Od samego początku funkcjonowania Internetu ludzie uznali, że ciekawym sposobem na wykorzystanie tej technologii jest rozgrywanie partii szachowych rozgrywanych na drugim krańcu świata. Już w latach 80-tych ubiegłego wieku powstały pierwsze technologie i protokoły pozwalające na rozgrywki poprzez sieć. Od początku obecnego wieku jest to rozrywka coraz popularniejsza i skupiająca ludzi z całego świata. Rozmaite kafejki internetowe oferują użytkownikom możliwość gry w szachy[25] zarówno z ludzkimi rywalami, jak i przeciwko komputerom szachowym. Użytkownicy posiadają do wyboru opcje wyszukiwania przeciwnika wg ustalonych przez siebie kryteriów. Preferencje graczy są różne - niektórzy chcą zagrać w ciągu godziny jedną partię, inni w tym czasie chcą ich skończyć kilkanaście. Cechą wspólną dla każdego użytkownika jest jednak praktyczne podejście do gry - muszą mieć możliwość sprawnego rozgrywania partii szachowej, podczas której rzadko przełączają się między oknami przeglądarki czy też odchodzą od komputera.

Badania dotyczące kafejek szachowych przeprowadziłem na podstawie najpopularniejszego w Polsce serwisu na platformie kurnik.pl. Mimo iż profesjonalni szachiści omijają ten serwis z daleka, tysiące mniej zaawansowanych graczy spędza na nim godziny swojego czasu. W ramach badań przeprowadziłem ankietę, w której wzięło udział kilkudziesięciu zawodników korzystających kafejki. Pierwszym wnioskiem z wyników ankiet jest fakt, iż szachiści niezbyt przejmują się szatą graficzną interfejsu. Dochodzi tutaj do paradoksu, że szachiści korzystający z tej aplikacji w dużych ilościach uznają ją za estetyczną, natomiast każda osoba z zewnątrz świata szachowego wyraźnie krytykowała wygląd. Jest to dość ważna obserwacja, gdyż można z niej wyciągnąć wnioski przy projektowaniu grafiki serwisu. Nie oznacza to, że sprawę wyglądu można całkowicie pominąć, jednak nie należy kłaść na nią najwyższego priorytetu. Ważną kwestią, o która ankietowani zostali zapytani, był sposób oznaczenia ostatniego posunięcia. Większość osób uznała, że sposób w jaki na kafejce kurnik jest zadowalający, jednak istnieje także spora liczba osób, które nie było zadowolone z prezentowanego rozwiązania. Inne serwisy rozwiązują ten problem w rozmaite sposoby. Na serwisie ICC korzystającym z programu BlitzIn ostatnie posunięcie oznaczone jest czerwonymi obwódkami wokół dwóch pól[26]. Podobny system jest używany w serwisie Chessbomb, o którym będę pisał w dalszym rozdziale. On także używa obwódek wokół pól, jedynie zaznacza je kolorem niebieskim. Kafejka Kurnik funkcjonuje w podobny sposób, jednak zamiast obwódek zmienia kolor całego pola na nieco ciemniejszy. Zupełnie inną filozofie prezentuje serwis PlayChess, który ostatnie posunięcie zaznacza za pomocą kolorowej strzałki prowadzącej od pola początkowego do końcowego.

Kwestia zapisu partii nie jest aż tak istotna, jakby mogło się to wstępnie wydawać. Użytkownicy po skończonej partii rzadko zastanawiają się, gdzie popełnili błąd, częściej zaś siadają natychmiast do następnej rozgrywki. Wśród ponad sześćdziesięciu uczestników ankiety tylko jedna osoba zwróciła uwagę na to, że w podstawowym wyglądzie okna nie ma

[25] A czasem także w gry podobne do szachów

[26] Startowym i końcowym polem figury która wykonała ostatnie posunięcie

zapisu. Pozostałe kafejki udostępniają zapis w podstawowym widoku, jednak najwyraźniej rozwiązanie przyjęte w kurniku spotyka się z aprobatą graczy.

Jako główną zaletę kafejki kurnik przedstawiana często była duża liczba osób dostępna do rozgrywki. Wynika z tego, że wskazanym jest, aby użytkownicy także mieli kontakt z innymi osobami korzystającymi ze strony. W przypadku kafejek szachowych, gdzie główną ideą jest rozgrywanie partii z innymi użytkownikami jest to bardziej oczywiste niż na pozostałych serwisach, jednak kontakt z innymi szachistami wydaje się wskazany dla każdej witryny internetowej.

Wgląd do świata kafejek szachowych jest istotny, nawet jeśli nie są one podobne do aplikacji służących do oglądania partii na żywo. Mają inne cele, inne priorytety i inne sposoby przedstawiania rozgrywki. Kafejka musi zapewnić użytkownikowi wygodę w znalezieniu dopasowanego przeciwnika, musi działać w czasie rzeczywistym i dostarczać bardziej zabawę niż edukację. W kafejce szachowej dużo ważniejsze jest przedstawienie czasu i ostatniego posunięcia, mniej ważny jest zaś dostęp do zapisu posunięć. Mimo to sposoby oznaczania szachownic i figur są wspólne dla obu zagadnień i dlatego wiedza z rozwiązań występujących w kafejkach szachowych można wykorzystać podczas tworzenia innych serwisów.

3.2 Programy szachowo-bazodanowe

Profesjonalni szachiści bardzo często korzystają z programów pomagających im zarówno w treningu, jak i wyszukiwaniu informacji o przeciwnikach. Jest to bardzo dobra grupa użytkowników do badań, gdyż najczęściej używają oni wybranych produktów w dłuższym okresie trwania swojej sportowej kariery. Ostatecznie przekłada się to na tysiące godzin spędzonych nad aplikacją, której zalety i wady poznają od podszewki. W przypadku tych programów sprawa się ma podobnie jak z kafejkami szachowymi - realizują one inne cele niż aplikacje służące relacji na żywo, jednak część rozwiązań może być wykrorzystana także w przeglądarce partii. Najważniejszą cechą tych programów jest wspieranie wyszukiwania partii po kluczowych danych (czyli czynność zupełnie niespotykaną zarówno w kafejkach szachowych, jak i w programach do przeglądania partii). Dodatkowo konieczną funkcją jest zaimplementowanie interfejsu do obsługi silników szachowych[27].

Obliczenia przeprowadzanie przez silniki szachowe są bardzo kosztowne dla zasobów komputera - zarówno pod względem zużycia procesora, jak i pamięci. Z tego powodu każdy program szachowy włącza silnik dopiero na wyraźne żądanie użytkownika. Doświadczony szachista najlepiej wie, w którym momencie partii należy włączyć program analizujący.[28] Programy po włączeniu pokazują kilka najlepszych według nich opcji dla danej pozycji[29] posortowanych względem oceny końcowej wyliczonej algorytmem MIN-MAX. Parametry uruchomienia silnika mogą podlegać modyfikacjom. Można modyfikować wielkość tablicy haszującej[30], można systemowo przydzielić procesowi wyższy priorytet. Wiadome jest, że z czasem to drzewo będzie miało tak wiele liści, iż później obliczenie każdego nowego półposunięcia będzie trwało mnóstwo czasu. Istnieją jednak sytuacje, gdy warto pozostawić dane

[27] Każdy szachista ma swój ulubiony silnik. Niektórzy używają różnych programów do oceny innych aspektów gry

[28] W naszej aplikacji nie będziemy mieli takiego luksusu - wszystko będzie się odbywało automatycznie

[29] Czasem zdarza się tak, że kilka ruchów jest równorzędnie dobrych. W innych przypadkach może się okazać, że różne posunięcia mogą doprowadzić do tej samej pozycji, a co za tym idzie oba warianty będą jednakowo ocenione

[30] Aczkolwiek jest to bardzo nieefektywny sposób. Wg Vasika Rajlicha, twórcy programu Rybka, podwojenie wielkości tablicy haszującej powoduje wzrost siły gry na poziomie 5 punktów ELO(obecnie komputery osiągają ranking rzędu 3200)

obliczenia nawet na kilka-kilkanaście godzin. Drużyny narodowe niektórych krajów, aby móc wzmocnić siłę obliczeń zainwestowały w kilkunastordzeniowe komputery przenośne, tylko po to, aby uzyskać przewagę nad konkurentami.[31]
Dwoma głównymi programami bazodanowymi są rozwijane od kilkunastu lat komercyjne ChessBase i Chess Assistant. Istnieją także darmowe programy, jak choćby ChessDB.[32]

W przypadku programów szachowych kluczowe znaczenie ma zapis partii. Jako że służą one do nauki, przebieg dotychczasowych ruchów jest niemal zawsze kluczowy dla edukacji. Przy przeglądaniu rozgrywki obok szachownicy zawsze wyraźnie widać zapis dotychczasowych ruchów. Więcej, zazwyczaj jest on tak ułożony, by nie trzeba było przewijać aby zobaczyć całą partię. Jest on zwyczajowo interaktywny - kliknięcie kursorem na posunięcie powinno automatycznie przenieść ekran partii do momentu, w którym ruch został wykonany.

3.3 Tablice końcówek, książki debiutowe, programy sędziowskie

Bardzo ciekawym aspektem programistycznym są tzw. **tablice końcówek**. Jest to bardzo współczesny typ programów - jeszcze kilkanaście lat temu większość komputerów była zbyt słaba, aby poradzić sobie z problemem eksplozji wykładniczej, która dotyczy tego problemu. Ogólnym celem tego programu jest opisanie w olbrzymiej bazie danych **wszystkich** pozycji o bardzo ograniczonej liczbie figur. Jak łatwo można się domyśleć, liczba takich pozycji rośnie wykładniczo względem ilości figur pozostałych na szachownicy. Dzięki zapisaniu całości w specjalnie do tego celu dostosowanej bazie danych możemy w czasie liniowym określić, czy dana pozycja jest remisowa czy też wygrana dla jednej ze stron(tutaj już nie ma mowy o przewadze - dla danej pozycji znamy już pełne drzewo MIN-MAX-owe, więc rezultatem **zawsze** jest wynik partii). Najbardziej znanym programem tego typu jest tzw. Nalimov (Nalomov tablebases), który w chwili obecnej zawiera niemal wszystkie pozycje posiadające sześć lub mniej figur. Obecnie trwają prace nad wprowadzaniem dodatkowej figury, jednakże

[31] Jeszcze dalej posunął się Bułgar, Veselin Topałow. Przygotowując się do meczu o Mistrzostwo Świata korzystał z najszybszego komputera w swoim kraju udostępnionego przez sofijski bank

[32] http://chessdb.sourceforge.net/

szacuje się, że dla siedmiu figur cała baza będzie zawierała około 1,2TB danych.

Książki debiutowe są bardziej wyzwaniem pod względem inżynierii oprogramowania i są częstym dodatkiem do silników grających. Posiadają one wprowadzoną wiedzę szachową na temat najbardziej popularnych otwarć. Wiedza ta nie jest generowana komputerowo[33], a umieszczana w niej jest ogólna wiedza szachowa dotycząca debiutów(często poznawana z dużą pomocą komputerów). Dzięki temu także wiedza zawarta w tych książkach jest bardziej „ludzka" - komputery czasem grają ruchy niezrozumiałe dla człowieka.

Osobną grupą programów są aplikacje sędziowskie. Aby móc sprawnie zarządzać zawodami sędziowie potrzebują aplikacji, które pomogą im w ustalaniu kojarzeń[34], liczeniu wyników, ustalaniu wyników rankingowych, norm na wyższe kategorie. Są to programy implementujące przepisy i algorytmy zawarte w kodeksach szachowych.[35] Jednym z najpopularniejszych programów tego typu jest polski Chess Arbiter[36]

3.4 Chessbomb

Serwis Chessbomb powstał i został udostępniony użytkownikom na początku 2009 roku. Służy on do przeglądania on-line partii szachowych **wraz z wbudowanym modułem oceniającym za pomocą silnika szachowego aktualną pozycję.** Owa funkcjonalność jest nowością - wcześniejsze przeglądarki on-line nie oferowały natychmiastowej analizy pozycji, co zmuszało użytkowników do korzystania ze swoich lokalnie zainstalowanych programów. Trzeba od razu podkreślić, że wcześniejsze rozwiązanie posiadało jedną przewagę nad sposobem udostępnionym przez ChessBomb. Użytkownik posiada możliwość poświęcenia na

[33]Dla debiutów, gdzie wciąż sa prawie wszystkie figury, do wariantów proponowanych przez komputer trzeba podchodzić z wyjątkową ostrożnością. Zdarzyło mi się przegrać partię, bo komputer się pomylił w swojej ocenie

[34]Ustalanie, kto z kim będzie grał w kolejnej partii

[35]Jak na przykład algorytm wyliczania rankingu oparty na rozkładzie normalnym zmiennej losowej

[36]http://www.chessarbiter.com/

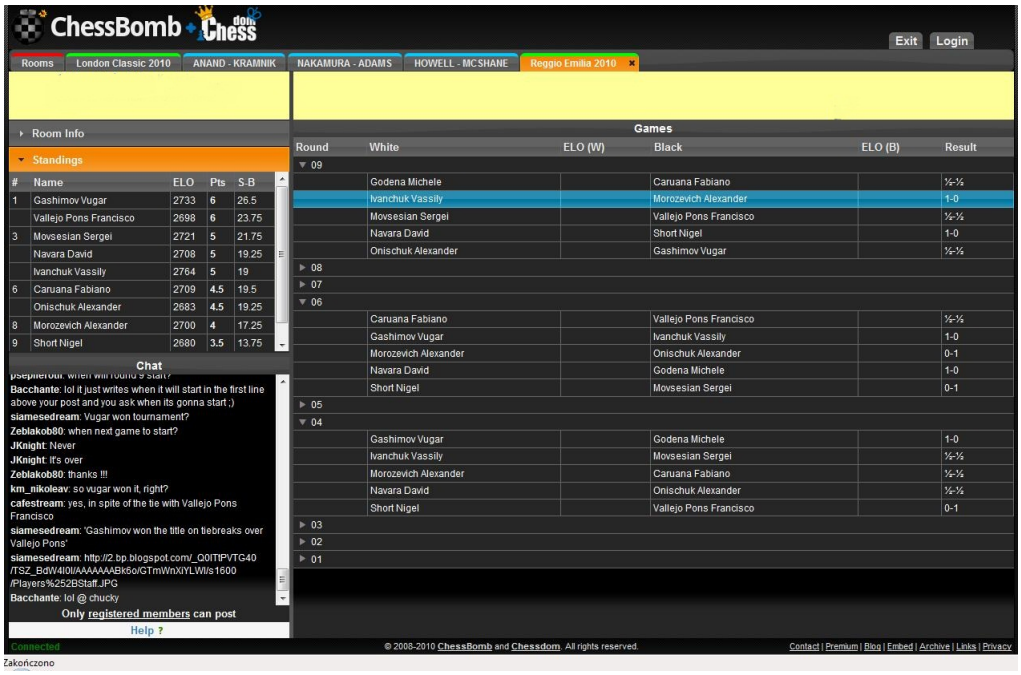
wyliczenie oceny dla danej pozycję całą moc swojego procesora i tyle czasu, ile on uzna za konieczne. Dlatego analizy dokonywane lokalnie są nie tylko wiarygodniejsze, ale także czasem udowadniają, że ocena po stronie serwera była błędna. Jeśli tworzymy aplikację, która ocenia pozycję po stronie serwera musimy się z tym pogodzić - nasze analizy i tak będą statystycznie dużo lepsze niż „ludzkie przemyślenia". Choć w czysto teoretycznym sensie nie będą one idealne, w praktyce będą miały bardzo duże zastosowanie. Jakie są zatem zalety uruchamiania silnika szachowego po stronie serwera?

- Obliczenia dostępne są zazwyczaj szybciej, niż w przypadku samodzielnego obsługiwania programu grającego (zwłaszcza jeśli śledzi się więcej jak jedną partię). Wypatrywanie nowych posunięć, wprowadzanie nowych posunięć, włączanie i wyłączenia silnika - każda z tych operacji jest nie tylko irytująca dla użytkownika, ale także i czasochłonna.
- Dużo łatwiej jest zarządzać obserwowaniem analiz, nie potrzeba przełączać okienek systemowych - wszystko mamy pod jednym widokiem
- Unika się błędów związanych ze złym wprowadzeniem posunięć (każdy krok pomiędzy wykonaniem posunięcia przez szachistę a wyświetleniem być może na drugim krańcu świata analizy tej pozycji jest zautomatyzowany)
- Można korzystać z analiz, nawet jeśli nie posiada się programu grającego[37]

Jako pierwsze rozwiązanie tego typu, ChessBomb nie jest pozbawiony wad. Testy przeprowadzone z użytkownikami wykazały jednak, iż większą część czynności wykonuje się na nim w sposób intuicyjny i nie przysparzający problemów. Serwis posiada dość niewielką liczbę funkcjonalności, jednak te, które są dostarczane, są relatywnie łatwe w użyciu nawet dla niedoświadczonego użytkownika. Takie wnioski wynikły zarówno z internetowych ankiet, jak i bezpośrednich testów z użytkownikami.

Wygląd interfejsu programu ChessBomb często uznawany jest za nieciekawy, tym bardziej że twórcy nie udostępnili żadnego sposobu jego zmiany. Za bardzo wygodny w użytkowaniu uznany jest system zakładek, za pomocą których można bez najmniejszych problemów przełączać się między oknami poszczególnych partii.

[37]Silniki szachowe są bardzo często bezpłatne. Inaczej jest z programami implementującymi interfejs do tych silników



3.5 Cechy wspólne dotychczasowych rozwiązań oraz ich ocena

Choć każdy z przedstawionych dotychczas programów zajmował się szachami, to każdy z nich skupiał się na innym aspekcie rozgrywki. Wynika z tego, że rozwiązania te nie mają wielu wspólnych elementów. W poniższych punktach zebrałem cechy łączące zaprezentowane programy.

1. **Niektóre standardy opisu partii**
   Każdy z programów i systemów posiada swój własny sposób zapisywania nagłówków partii. Programy bazodanowe skupiają się na zapisaniu jak największej liczby informacji, aby umożliwić wyszukiwanie wg większej liczby kryteriów. Standardem jest zapisywanie danych graczy, przy czym kluczem i pierszą wyświetlaną wiadomością jest niemal zawsze nazwisko zawodnika. Dodatkowym zwyczajem jest wyświetlanie zawodnika grającego białymi przed zawodnikiem grającym czarnymi.

2. **Graficzne przedstawianie figur**
   Choć oczywiście można zamiast piktogramów do wyświetlania pozycji używać małych i wielkich liter, to dla wygody użytkowników figury szachowe zawsze przedstawiane są w postaci graficznej. Dodatkowo figury białych i czarnych różnią się jedynie kolorem - ich kształty pozostają takie same. Nie ma także różnicy między takimi samymi figrami tej samej strony.

3. **Wyświetlanie białych figur w kolorze białym oraz czarnych w pozostałych kolorach**
   Choć istnieje wiele interfejsów graficznych przedstawiających partię szachową, kolory wybierane do oznaczania figur w każdym rozwiązaniu są standardowe. Białe figury najczęściej przedstawiane są w różnych odcieniach bieli, dla czarnych figur jest zaś większa dowolność. Mogą być to klasycznie czarne, mogą być ciemnobrązowe, ale zdarzają się także ciemnoniebieskie i ciemnozielone.

4. **Umieszczanie szachownicy w środku ekranu**
   Niezależnie od rozmaitych celów aplikacji, szachownica przedstawiająca rozgrywkę za-

wsze ukazuje się w centrum ekranu. Czy jest to kafejka szachowa, czy jest to baza partii czy witryna do oglądania szachów na żywo - szachownica zawsze zajmuje centralną pozycję na stronie. Dodatkowo warto zauważyć, że w przypadku gdy sama wielkość szachownicy nie jest modyfikowalna, to szachownica zajmuje między $\frac{1}{3}$ a $\frac{2}{3}$ szerokości okna (chyba że ekran jest wyjątkowo nieproporcjonalny - trzeba pamiętać, że szachownica zawsze jest kwadratem)

5. **Obsługa i generowanie plików PGN**
   Zdecydowana większość programów szachowych potrafi zarządzać plikami PGN. Choć programy bazodanowe najczęściej operują na swoich własnych strukturach, to zawsze dodatkowo obsługują format PGN. Jeszcze częściej zjawisko to widać w przypadku przeglądarek graficznych plików - w większości przypadków PGN to jedyny obsługiwany format plików. W przypadku kafejek szachowych sytuacja jest odwrotna - lepsze serwisy generują pliki PGN na podstawie rozegranych partii.

6. **Zablokowane wykonywanie niepoprawnych posunięć**
   Niezależnie od tego, czy program obsługuje transmisję na żywo, czy też wyświetla partię z pliku - zawsze dokonywana jest weryfikacja danych. Programy nie pozwalają na wykonanie niepoprawnego posunięcia. Także standardem jest brak obsługi tego błędu. Powód tego jest dość trywialny. Sytuacja taka jest niezwykle rzadka i nie istnieją znane algorytmy zaradzenia temu problempowi. Więcej, można łatwo udowodnić, że nie istnieje uniwersalny sposób, który zawsze skutecznie naprawi błędne dane.

7. **Przemieszczanie figur z użyciem klawiatury**
   W dzisiejszych czasach coraz bardziej odchodzi się od korzystania z klawiatury w celach nawigacji. Programy szachowe stanowią jednak wyjątek od tej reguły. Standardem jest, że można przeglądać posunięcia korzystając z klawiaturowych strzałek. Jest to podyktowane wygodą - kiedy partia szachowa składa się kilkudziesięciu półruchów nikomu nie chce się tyle razy naciskać na ikonkę przewijania. Klawisz strzałki zaś można przytrzymać, co wyraźnie oszczędza czas użytkownika.

8. **Przedstawianie czasu**
   Zarówno w kaefjkach, jak i w transmisjach on-line trzeba pokazywać bieżące zużycie czasu zawodników. Wszędzie korzysta się ze standardu pokazywania, jak wiele czasu zostało zawodnikom. Jest to naturalne podejście (znów - takie samo jest w przypadku fizycznych zegarów szachowych) i każdy interfejs realizuje go w podobny sposób[38]. W tym przypadku można rozróżnić transmisje i grę na kafejkach, gdyż w transmisjach samo opóźnienie może być kilkudziesięciosekundowe[39], w kafejkach internetowych zaś nawet półsekundowe opóźnienie może być kluczowe[40]. Dlatego też w kafejkach często podaje się czas z dokładnością do dziesiątych części sekundy, zaś przy transmisjach dokładność tego stopnia jest absolutnie niepotrzebna.

[38] Nigdy nie podaje się czasu zużytego, gdyż w przypadku zakończenia partii nie ma on wpływu na żaden aspekt wyniku

[39] Z powodów podanych w rozdziale dotyczących szachownic elektronicznych

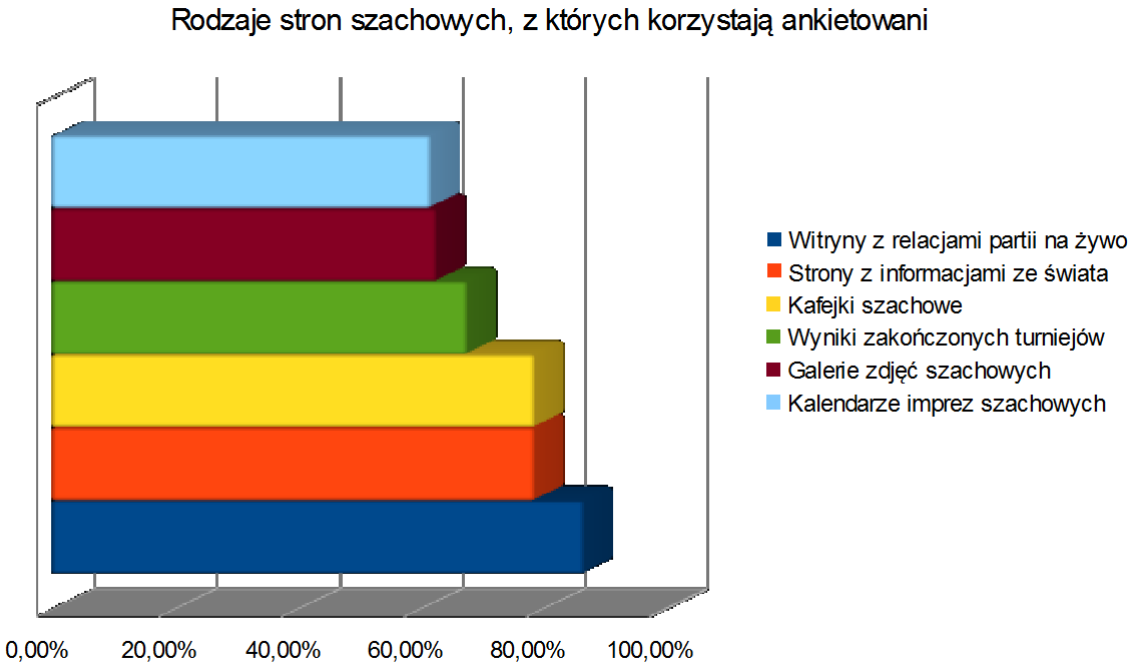
[40] Wiele osób grywa partie, w których na całość przeznaczona jest zaledwie jedna minuta

4 Projektowanie własnego interfejsu

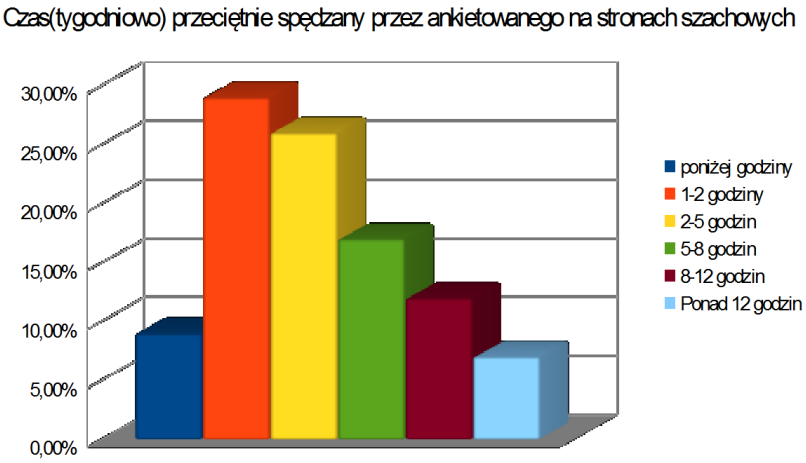
Nadszedł czas, aby przejść od etapu oceniania gotowych rozwiązań do zaproponowania własnych. Moim założeniem było stworzenie serwisu podobnego w głównej funkcjonalności do strony Chessbomb, jednak realizującego także inne funkcje.

4.1 Wyniki ankiet

Przed rozpoczęciem projektowania własnego programu skierowałem pytania do szerokiej grupy użytkowników pewnej strony szachowej. Odzew był bardzo duży i ostatecznie w badaniach wzięło udział, aż 130 uczestników. Część pytań dotyczyła każdej z tych osób, niektóre z nich jedynie części tej grupy. W każdym pytaniu, poza pytaniem o czas spędzany na stronach szachowych, można było wybrać więcej niż jedną odpowiedź.

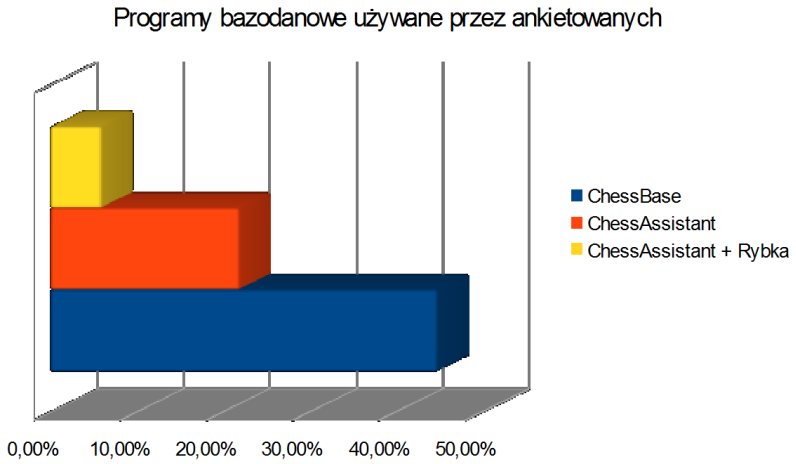


Pierwsze pytanie było najbardziej ogólne ze wszystkich. Uczestnicy zostali zapytanie o rodzaje witryn szachowych, na których zdarza im się przebywać. Pierwszym, najbardziej oczywistym wnioskiem z przedstawionych na diagramie wyników jest fakt, iż każdy z rodzajów stron szachowych znajduje wielu sympatyków (nawet najmniej popularna opcja zdobyła ponad 50% głosów). Z tego wynika także kolejny wniosek - większość internautów korzysta ze stron o różnych zastosowaniach. Jako iż niewiele jest serwisów oferujących więcej niż jedną z tych funkcji, przypuszczać można, że odwiedzają jednocześnie kilka różnych witryn. Zwraca uwagę także główne zainteresowanie ankietowanych - czterech na pięciu z nich przyznaje się, do oglądania na żywo zawodów szachowych poprzez relacje internetowe.



Aby uzyskać lepszy obraz przeciętnego szachowego internauty zadałem także pytanie na temat czasu, który przeciętnie w ciągu tygodnia przeznaczają na przeglądanie witryn szachowych. Z wyników odpowiedzi na to pytanie można wnioskować, że większość osób korzysta z tych serwisów przez krótkie okresy czasów, przeznaczając na to średnio kilkanaście do kilkudziesięciu minut dziennie. Warto jednak zwrócić uwagę, że bardzo mała jest grupa osób, które na strony szachowe wchodzą niezwykle rzadko - podczas gdy istnieją internauci spędzający nad szachami ponad godzinę dziennie.

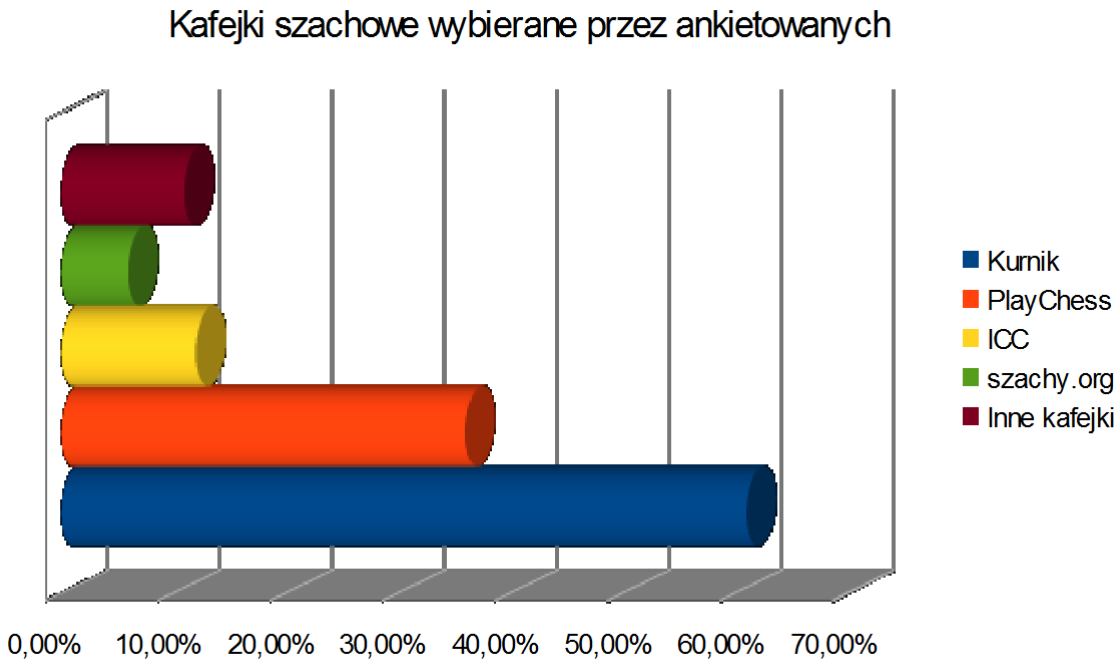
Do zaprezentowanej powyżej tej ankiety dodać jeszcze komentarz niezależny od jej wyników. Średnio dwa-trzy razy w roku odbywają się ważne turnieje, w których występuje reprezentacja Polski. Strony internetowe związane z szachami przeżywają wówczas prawdziwe oblężenie - przeciętna liczba dziennych odsłon zwiększa się czasem nawet kilkukrotnie. Wówczas użytkownikom zdarza się spędzać nawet kilka godzin na różnych serwisach oferujących transmisję z tych rozgrywek.



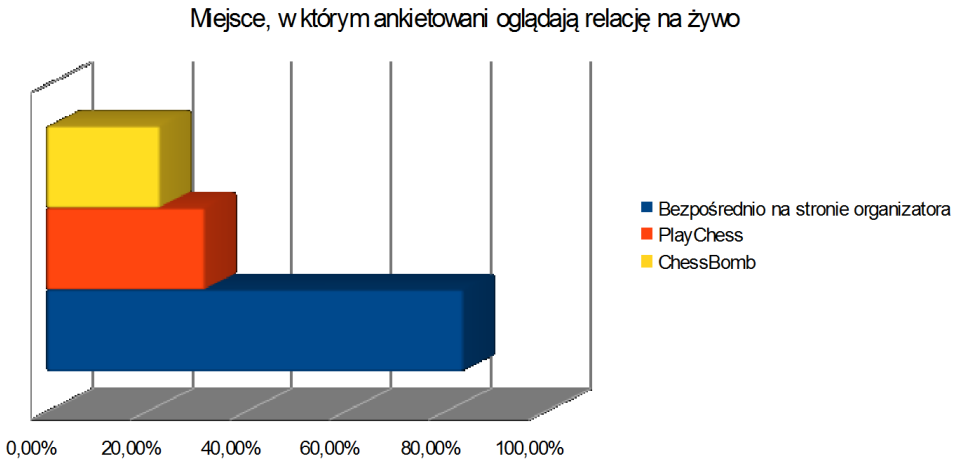
Kolejne pytanie w ankiecie dotyczyło użycia programów bazodanowych zawierających partie szachowe. Aplikacje posiadające tę funkcję są płatne i implikuje to wyraźnie mniejszą liczbę użytkowników programów w porównaniu do darmowych witryn internetowych. Najbardziej popularnym rozwiązaniem jest program ChessBase, którego posiada niemal połowa ankietowanych. Dużo mniejszym zainteresowaniem cieszy się odwieczny konkurent ChessBase'a - Chess Assistant[41] Wielkiej popularności nie zdobył także najnowszy produkt na rynku,

[41] Co ciekawe, w Polsce profesjonalni szachiści częściej używają właśnie tego programu.

który łączy w sobie możliwości ChessAssistanta i jednego z najsilniejszych programów szachowych, Rybki.



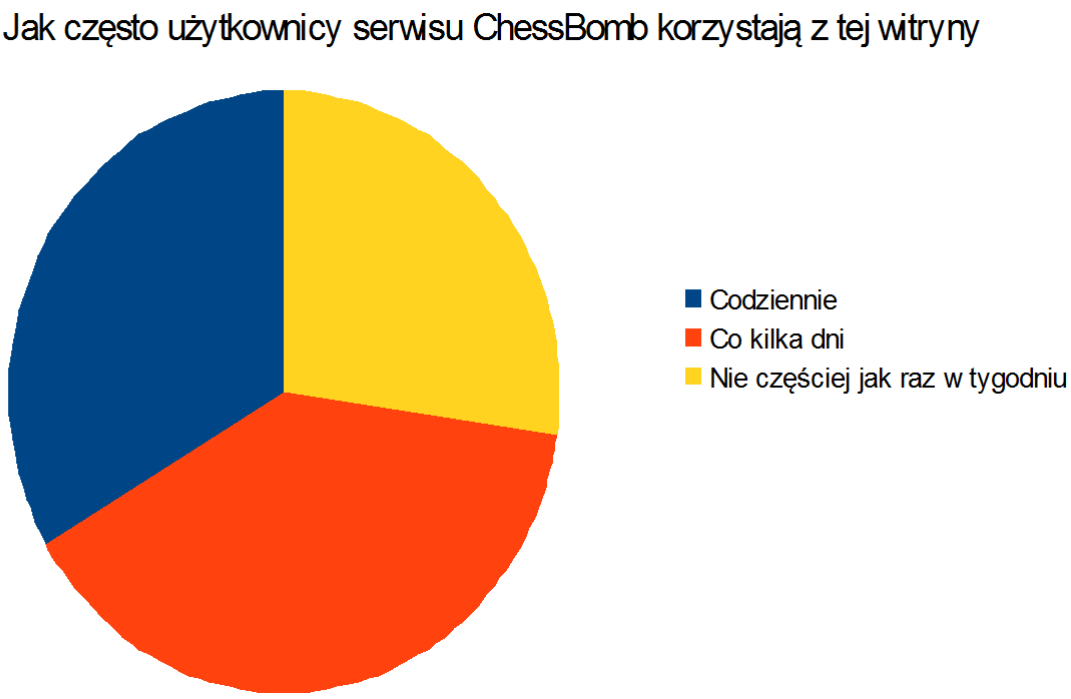
Z pierwszej ankiety wynikało, iż prawie 80% internautów korzysta z Internetu do rozgrywania partii szachowych. Pytanie o wskazanie kafejek internetowych, z których korzystają użytkownicy dało bardzo czytelne wyniki jeśli chodzi o popularność różnych rozwiązań. Zdecydowanie najwięcej głosów zdobył serwis Kurnik, do używania którego przyznało się dwóch na trzech ankietowanych. Niemal połowę mniej głosów zdobył drugi w rankingu popularności komercyjny serwis PlayChess uruchamiany przez program firmy ChessBase. Bardzo niewielu zwolenników miały zaś rozwiązania renomowane lecz zagraniczne (Internet Chess Center) oraz kiedyś bardzo popularne, polskie szachy.org[42].



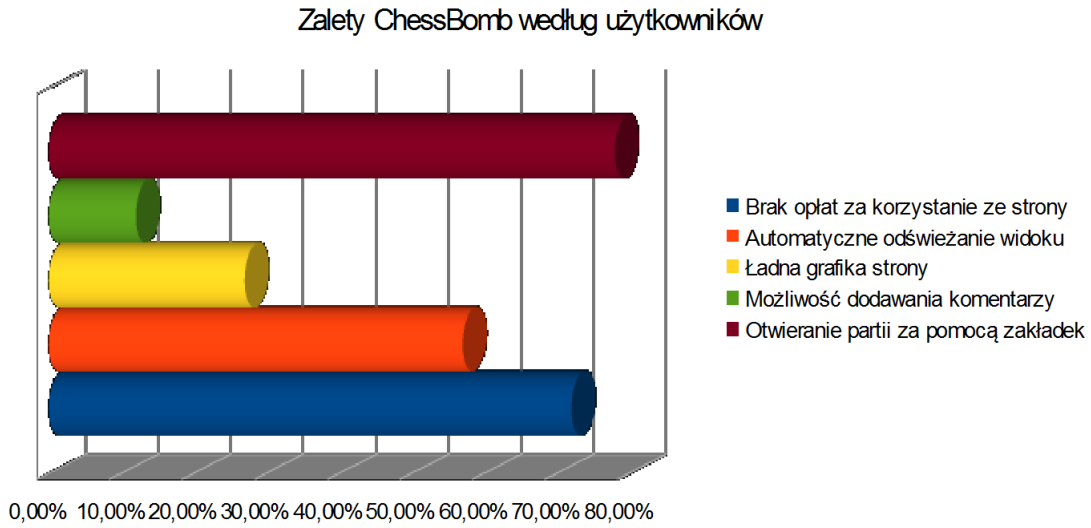
Aby mieć możliwość oglądania partii szachowych przez Internet trzeba wybrać jedną z dwóch dróg. Najbardziej klasycznym rozwiązaniem jest otworzenie strony turnieju i odnale-

[42]Kafejka stworzona przy XIV L.O. we Wrocławiu przez Maurycego Prodeusa i Wojciecha Samotija.

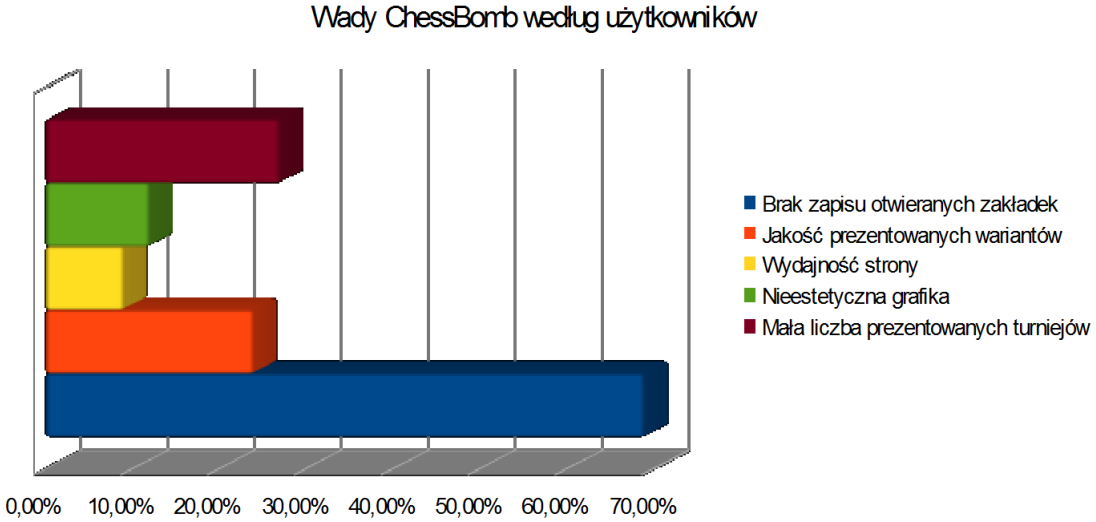
zienie na niej linku do bezpośredniej transmisji partii. Drugim wyjściem jest uruchomienie program lub witryny, która „zbiera" takie transmisje. Choć mogłoby się wydawać, że drugie rozwiązanie jest wygodniejsze dla użytkowników, to wyniki ankiet świadczą o czymś wręcz przeciwnym. Zdecydowana większość użytkowników decyduje się na korzystanie z bezpośredniej transmisji na stronie organizatora, jedynie część używa stron lub programów niezależnych od tego konkretnego turnieju. Dlaczego tak się dzieje? Przypuszczam, że wynika to z dwóch faktów - część z tych rozwiązań jest płatna (PlayChess), inne zaś istnieją zbyt krótko, by zdobyć uznanie większości użytkowników(ChessBomb)



Pytanie dotyczące serwisu ChessBomb zostały przeprowadzone na mniejszej próbce użytkowników - spytani zostali tylko ci, którzy korzystają z tego serwisu. Pierwszym zagadnieniem, o które zostali oni zapytani była częstość wchodzenia na tę stronę. Widać wyraźnie, że nie ma jednolitego trendu - jest sporo grupa osób, która odwiedza tę stronę codziennie, ale podobnie wiele osób zagląda na nią bardzo sporadycznie. Także grupa osób, które odwiedzają tę witrynę raz na kilka dni nie wyróżnia się swoją wielkością i jest porównywalna z bardziej skrajnymi odpowiedziami.

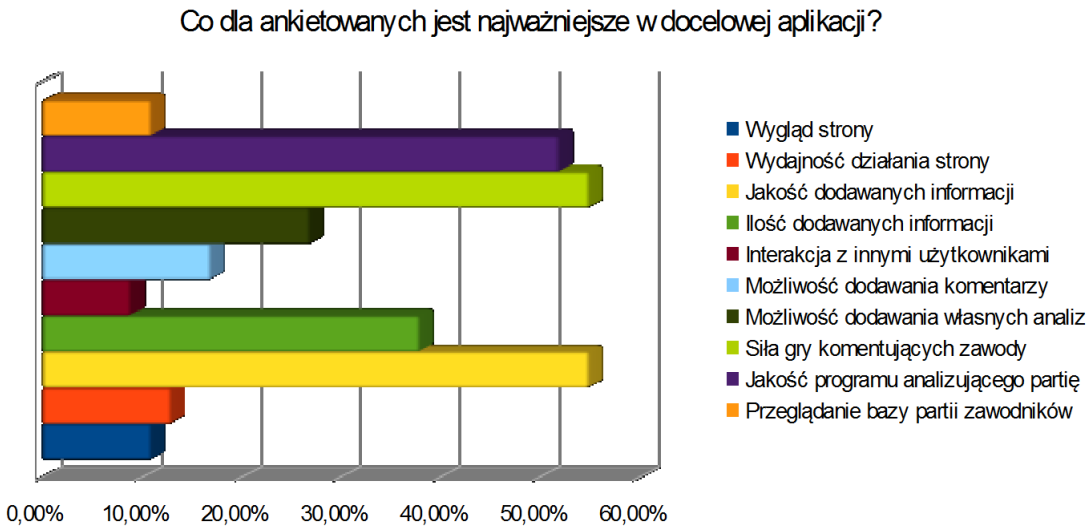


W celu późniejszego odniesienia mojej aplikacji spytałem się użytkowników, jakie elementy uznają za największy atut ChessBomb. Wyraźnie widać, iż najbardziej użyteczną funkcję oferowaną przez tę witrynę jest sposób otwierania okien z nowymi partiami poprzez system zakładek. Użytkownicy chwalą sobie także automatyczne odświeżanie okien, kiedy jest zagrane nowe posunięcie. Fakt, iż korzystanie z większości funkcji jest darmowy także nie umknął uwadze ankietowanym. Niewiele osób natomiast uznało, że głównymi zaletami witryny są wygląd strony lub też możliwość dodawania komentarzy słownych



Nie można mówić o rzetelnej ocenie serwisu, jeśli nie opisuje się także złych rozwiązań w nim zastosowanych. W przypadku ChessBomb zdecydowanie najwięcej osób zwróciło uwagę na brak zapisywania stanu przeglądania partii. ChessBomb nie oferuje żadnego sposobu na przechowywanie informacji(nawet w ciasteczkach), zatem przypadkowe zamknięcie okna przeglądarki powoduje, że później raz jeszcze trzeba otwierać wszystkie dotychczas obserwowane okna. Pozostałe wady nie przeszkadzają użytkownikom - zaledwie jeden na czterech uważa, że wadą jest zbyt mała partia transmitowanych zawodów(dodatkowo ta w ostatnich miesiącach wyraźnie się zwiększyła). Podobna liczba osób uważa, że warianty prezentowane przez komputer są zbyt mało wiarygodne i serwer powinien poświęcić więcej zasobów na lepsze obliczenie wariantów. Tylko kilka osób uznało, że strona ChessBomb jest nieestetyczna

lub mało wydajna.



Końcowe pytanie zadałem znów szerszej grupie osób. Zostały one spytane, czego oczekiwałyby po nowo powstającej witrynie. Zaznaczyłem, że będzie ona realizowała podobną funkcjonalność jest strona ChessBomb, wzbogacone o informacje na żywo ze świata szachów. Najważniejsze dla ankietowanych okazały się następujące kategorie: jakość dodawanych informacji oraz siła zawodników komentujących partie szachowe. Bardzo ważne dla przyszłych użytkowników okazała się jakość wariantów proponowanych przez komputer. Dalsze miejsca z wyraźnie już mniejszą liczbą głosów zdobyły ilość wyświetlanych informacji oraz możliwość dodawania własnych partii, które zostaną przeanalizowane przez komputer. Dużo mniejsze znaczenie miały dla użytkowników takie kwestie jak wygląd serwisu i jego wydajność, a także możliwość interakcji między użytkownikami.

4.2 Rodzaje użytkowników

Nie sposób myśleć o programie jako o stuprocentowo automatycznej witrynie. Niektóre czynności z konieczności będą musiały być wykonywane przez administratora strony. Oczywiście należy dążyć do zminimalizowania nakładu pracy ludzkiej poprzez maksymalne ułatwienie i automatyzację czynności, jednak najprostsze operacje będą musiały być wykonywane przez człowieke. Ogólnie zaplanowałem podział użytkowników na następujące kategorie:

- Administratorzy strony
- Moderatorzy strony
- Moderatorzy treści
- Zarejestrowani użytkownicy strony
- Niezarejestrowani użytkownicy strony

O funkcjonalnościach dostępnych dla użytkowników różnego typu piszę w rozdziale "Implementacja programu"

4.3 Przypadki i scenariusze użycia oraz papierowy interfejs

4.3.1 Scenariusze użycia

Scenariusze użycia są przydatnymi narzędziami podczas projektowania interfejsów. Pomagają one wyobrazić sobie kontekst, w jakim będzie używana aplikacja. Dzięki temu można lepiej zrealizować funkcje, tak aby przykładowym użytkownikom dobrze się z nich korzystało. Przykladowe scenariusze użycia

Scenariusz #1

John Doe siedzi w swoim pokoju o godzinie 8:30 rano w niedzielę. Normalnie jeszcze by spał, ale jest relacja z Bardzo Ważnych Zawodów. Uruchomił stronę www, która to uruchomiła witrynę, na której prowadzona jest relacja na żywo. Na stronie głównej zauważył informację, że tego dnia nie będzie komentarza ludzkiego, jedynie automatyczny. John Doe uruchamia witrynę, która wyświetla mu listę wszystkich turniejów dostarczanych przez stronę internetową. Lista jest posortowana chronologicznie, więc wybiera z niej Bardzo Ważne Zawody. Pokazuje mu się lista aktualnie granych partii. Lista jest posortowana względem szachownic, jednak John wybiera opcję zmiany na sortowanie alfabetyczne. Wybiera z listy partię swojego ulubieńca, Seana Smitha, który gra z Andriejem Kałmukowem. W wyniku wybrania otwiera się nowa zakładka z partię Smith-Kałmukow. John Doe widzi, iż zostało wykonanych już 15 posunięć i białe zużyły 20 minut, czarne zaś 10. Z przebiegu partii wybiera pierwsze posunięcie i wybierając opcję pokaż następny ruch przechodzi do końca partii. Widzi ocenę pozycji po najlepszym ruchu Smitha i dwukrotnie klikając w wariant wyświetla go w postaci graficznej.

Scenariusz #2

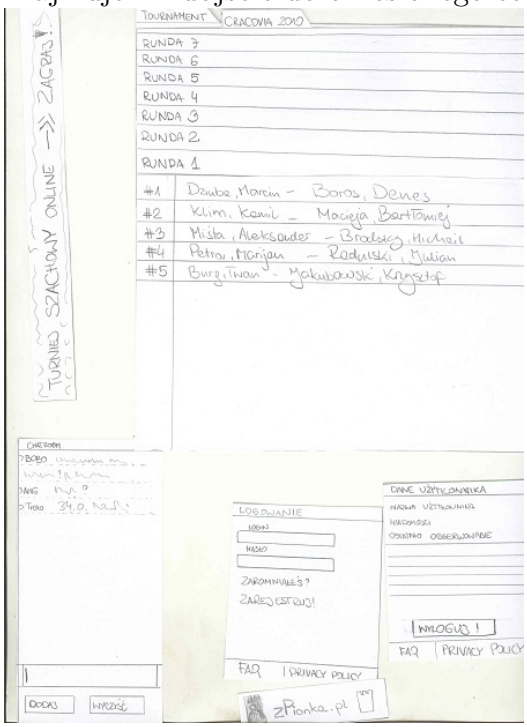
Michael Collins siada wieczorem do komputera, aby zobaczyć wyniki z dzisiejszej rundy Bardzo Ważnego Turnieju. Uruchamia stronę internetową, po czym wchodzi do opcji logowania. Wpisuje swoje dane i bezproblemowo loguje się na serwer. Witryna wyświetla mu listę turniejów, po czy Michael wybiera Bardzo Ważny Turniej. Wyświetliła mu się lista partii z wpisanymi do niej wynikami. Michael otwiera zakładki z partiami z sześciu pierwszych szachownic i przegląda każdą z nich osobno. Po przejrzeniu partii zamyka zakładkę. Po przejrzeniu wszystkich partii pozostaje w zakładce turnieju i w „shoutboxie" pisze komentarz na temat rundy. Po wykonaniu tej czynności wylogowuje się i zamyka przeglądarkę.

Scenariusz #3

Jack Wilshere przychodzi do komputera około godziny 12. Nie włącza niczego, gdyż zostawił przeglądarkę włączoną poprzedniej nocy. Widzi, że może kliknąć na pole oznaczające nową rundę i wyświetli mu się lista partii. Niektóre partie się zakończyły i są wpisane ich wyniki. Wchodzi na partię z pierwszej szachownicy i widzi, że poza posunięciami jest w zapisie partii komentarz słowny. Jack odchodzi od komputera i wraca po kwadransie z gorącą kawą. Widzi natychmiast, że zostało zagranych kilka ruchów, bo mimo iż nie dotknął ani myszki, ani klawiatury, pozycja się zmieniła. Po kolejnych pięciu minutach widzi wpisany wynik partii – zakończyła się remisem.

4.3.2 Papierowy prototyp

Papierowy prototyp, jak sama nazwa wskazuje, jest szablonem wyglądu interfejsu stworzonym za pomocą przyrządów biurowych. Mimo iż wydaje się to nieco nieprofesjonalnym podejściem, pozwala on na szybkie i mało kosztowne przetestowanie podstawowych użyteczności interfejsu. Tworząc taki schemat twórca zupełnie odcina się od sposobu w jaki tworzy interfejs i skupia się na samych funkcjonalnościach. W wyniku temu można szybko znaleźć największe problemy dotyczące użyteczności programu.
Papierowy prototyp pokazuje wszystkie najważniejsze elementy interfejsu, pozwalając testowanej osobie zobaczyć, jaki układ informacji będzie prezentowany na stronie. Uczestnicy testów dostali listę zadań do wykonania, podczas gdy obserwoałem sposób i mierzyłem czas jaki zajmuje im dojście do określonego celu.



Lista i kolejność zadań była następująca:

1. Zarejestrować się na stronie. Następnie znaleźć okno logowania,zalogować się na stronie. Na zakończenie zadania wylogować się z systemu.

2. Zalogować się na stronie, znaleźć listę turniejów i wybrać z niej turniej, którego data końcowa jest najstarsza. Napisać komentarz na chacie tego turnieju.

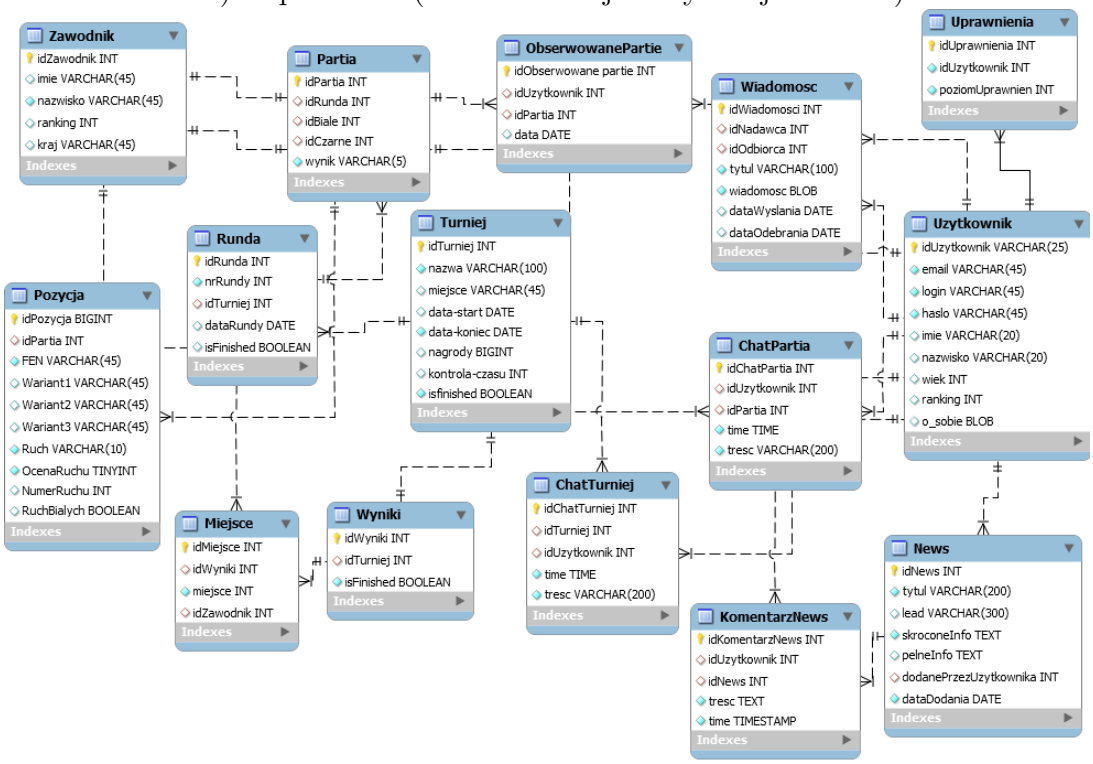
3. Pozostając w tym turnieju znaleźć partię z pierwszej rundy graną na pierwszej szachownicy. Wyświetlić ją w głównym oknie.

4. Przejrzeć partię ruch po ruchu. Następnie wrócić do początku rozgrywki. Przeczytać dane dotyczące zawodników

Uczestnicy testów byli zachęcani do wygłaszania swoich uwag i propozycji. Nie było także ustalonego limitu czasu na wykonanie danej czynności. Jako grupę testową użyłem szachistów w wieku 15-40 lat - docelową grupę użytkowników programu. Byli wśród nich zarówno doświadczeni użytkownicy interfejsów szachowych, jak i niedoświadczeni szachiści

4.4 Schemat bazy danych

Zanim rozpocząłem pracę nad projektem, stworzyłem wstępny schemat bazy danych, na której będzie opierała się aplikacja. Tabele zostały podzielone na dwie podstawowe grupy - związane z użytkownikiem i związane z partiami szachowymi. Połączone są ze sobą za pomocą kilku tabel, które będą miały za zadanie służyć przechowywaniu informacji o komentarzach użytkowników dotyczących różnych aspektów partii szachowych. Standardowo każda klasa posiada trzy pola - unikalne pole id, będącego kluczem głównym, oraz pola created_at(data utworzenia rekordu) i updated_at(data ostatniej modyfikacji rekordu)



5 Implementacja programu

5.1 Technologia

Zdecydowałem się pisać program w coraz modniejszej w ostatnich latach technologii Ruby on Rails. Choć do korzystania w pełni z możliwości udostępnianych przez biblioteki musiałem także korzystać z języków Java i JavaScript oraz ze znaczników HTML, to rdzeń aplikacji jest napisany w obiektowym języku Ruby.
O wyborze technologii, w której implementowałem aplikację zadecydowały dwie rzeczy. Pierwszy argument był programistyczny: Ruby on Rails jest technologią zwinną(agile) wspierającą wzorzec MVC, który bardzo dobrze pracuje w przypadku serwisów webowych. Drugi argument był zupełnie inny: chciałem poprzez pisanie pracy magisterskiej nauczyć się nowej technologii. Wcześniej nie miałem do czynienia ani z językiem Ruby, ani z frameworkiem Rails.

5.1.1 Ruby on Rails

„Ruby on Rails jest przełomem w dziedzinie programowania aplikacji internetowych. Potężne aplikacje, których tworzenie do tej pory zabierało tygodnie czy miesiące, są teraz tworzone dosłownie w kilka dni." -Tim O'Reilly, Założyciel O'Reilly Media[43]

Język Ruby powstał w roku 1995, stworzony przez Yukihiro "Matza" Matsumoto. Jest w pełni obiektowym i typowanym dynamicznie językiem bazującym na wcześniejszych rozwiązaniach, takich jak CLU, Eiffel, Lisp, Perl, Python czy Smalltalk Ruby cechuje się dużą zwięzłością kodu, przy posiadaniu czytelnej składni. Co ciekawe, w Ruby można modyfikować każdą klasę, włączywszy w to nawet podstawowe klasy systemowe (np. klasę String). Jedną z głównych zalet języka jest framework Ruby on Rails, który jest świetnym narzędziem do tworzenia aplikacji webowych

Ruby on Rails powstał jako niezależny projekt stworzony głównie przez duńskiego programistę Davida Heinemeiera Hanssona. Ruby on Rails jest narzędziem do szybkiego tworzenia aplikacji webowych **opartych na architekturze Model-View-Controller** przy zachowaniu zasady DRY[44] oraz reguły Convention over Configuration[45]. Jedną z większych zalet Ruby on Rails jest fakt, że posiada bardzo użyteczny moduł ActiveRecord oparty na ORM[46], który pozwala na tworzenie modeli w architekturze MVC niezależnych od faktycznej bazy danych. Kolejna zaletą zaletą tego frameworku jest dostępność świeżo rozwijanych i użytecznych bibliotek. Railsy zdobyły popularność także dzięki swojej początkowej prostocie - łatwo stworzyć podstawową witrynę w mniej niż kwadrans. Prędkość kodowania jest wspierana przez różnego rodzaju generatory kodu lub programy do zarządzania migracjami z bazą danych.

Do funkcjonowania witryn sieciowych potrzebny jest serwer, który będzie obsługiwał naszą aplikacje. W wersji deweloperskiej jest to najczęściej serwer WEBrick, gdyż jest domyślną częścią środowiska Rails. W fazie produkcyjnej częściej stosuje się albo serwera Mongrel(lub też całe grono serwerów Mongrel), albo Apache'a z modułem Phusion Passenger.

[43] O'Reilly Media to uznana amerykańska kompania medialna
[44] Don't Repeat Yourself - unikaj powtarzania kodu
[45] Minimalna początkowa konfiguracja zastąpiona gotowymi wzorcami
[46] Object-Relational mapping

5.1.2 JRuby on Rails

Choć Ruby jest już bardzo rozwiniętym językiem, czasami przydaje się jednak możliwość wykorzystania kodu napisanego w innych językach programowania. Takim rozwiązaniem jest JRuby, który pozwala na uruchomienie kodu Ruby na JVM (wirtualna maszyna Javy) oraz na korzystanie z klas napisanych w Javie. Podobnie jest samo Ruby on Rails, został on stworzony przez grupę programistów, którzy chcieli stworzyć jeszcze bardziej użyteczne narzędzie do tworzenia stron webowych. Dzięki bibliotekom JRuby można umieszczać strony napisane we frameworku Rails na urządzeniach przenośnych. Problemem z JRuby jest jedynie lekkie opóźnienie do klasycznego Ruby, najnowsze funkcje z frameworka Rails nie zawsze od razu działają pod JRuby. Korzystanie z klas Javy w Ruby jest proste - po zainstalowaniu JRuby wystarczy zaimportować klasy Javy za pomocą metod require lub include_class i korzystać z nich jak z klas z Ruby. Co więcej, JRuby automatycznie konwertuje standardy nazewnicze - funkcję toString przekształci na to_string[47]. Trzeba jedynie uważać na sytuację, gdy zarówno w Rubym, jak i w Javie są pliki o tej samej nazwie (wówczas najlepiej przekształcić nazwę jednej z klas z pomocą odpowiedniej metody umieszczanej w kontrolerze aplikacji). Trzeba także pamiętać, że przy uruchamianiu aplikacji napisanej w JRubym trzeba uruchomić serwer Railsów z katalogu JRuby.

5.1.3 Dodatkowe biblioteki

W programie wykorzystywałem gotowe biblioteki. Główną z nich jest **Chesspresso**[48], bardzo obszerna biblioteka napisana w Javie na licencji LGPL, służąca do parsowania i interpretowania plików szachowych PGN, a także interpretowania i tworzenia notacji FEN. Implementuje ona większość przydatnych funkcji potrzebnych do zarządzania partiami szachowymi, jej jedynym mankamentem jest nieistniejąca dokumentacja, która wymaga od programisty przynajmniej niewielkiego zagłębienia się w kod biblioteki i zdaniu się czasem na swoją intuicję.

Kolejną biblioteką jest nieznacznie zmodyfikowana na potrzeby pracy JavaScriptowa **jChess**[49] oprta na znanej bibliotece jQuery. Wizualnie prezentuje ona partię szachową, a także udostępnia funkcje pozwalające na nawigację. Dzięki nim można, z pomocą przycisków lub klawiatury, wykonywać posunięcia zarówno do przodu, jak i do tyłu.

Kolejną biblioteką jest **paperclip**[50] - plugin napisany w Rubym służący do wygodnego importowania plików na serwer, który można łatwo zintegrować z wymienionym przed chwilą jQuery. Wykorzystuję go przy procedurze zapisywania w bazie partii szachowej wczytywanej z pliku.

5.1.4 Programowanie zwinne

We wstępie do rozdziału wprowadziłem termin **programowanie zwinne**(Agile software development). Co kryje się pod tym terminem? Pod nazwą "metodyki zwinne" mieści się grupa metodyk opartych na programowaniu iteracyjnym. Różni się ona od standardowych metod tworzenia oprogramowania dużo mniejszą formalizmem i większą reakcją na potrzeby użytkownika. Jej zaletami są szybkość tworzenia działającego kodu i szybka reakcja na następujące zmiany. Jej słabością jest niemożliwość zastosowania w przypadku dużych projektów

[47] Choć dosłowne przepisanie nazwy funkcji też osiągnie zamierzony efekt.
[48] http://www.chesspresso.org/
[49] https://github.com/bmarini/jchess
[50] https://github.com/thoughtbot/paperclip

programistycznych, gdzie kontakt między poszczególnymi programistami jest utrudniony, a brak dokumentacji technicznej jest już realnym problemem.
"Rails skupia się na ludziach i interakcjach między nimi. Nie znajdziesz tu rozbudowanych pakietów narzędziowych, skomplikowanej konfiguracji, złożonych procesów. Są tylko małe grupki programistów, ich ulubione edytory, trochę kodu w Rubym. Rezultatem jest większa czytelność relacji; wyniki pracy programistów są natychmiast widoczne dla klienta. **Istotą tego pomysłu jest interaktywność**"[1]

Programowanie zwinne szczególnie nadaje się w przypadku stron internetowych, gdzie od samego początku pracy można współpracować z klientem, który bez posiadania specjalistycznej wiedzy może wyrazić swoją opinię na temat wyniku prac. Można wtedy co każdą iterację wprowadzać modyfikacje do przyszłego planu tworzenia aplikacji, uwzględniając zarówno aspekty, które wyszły na jaw w trakcie dotychczasowego pisania kodu, jak i zdanie oraz opinie klienta aplikacji. Taka metodyka jest też bardzo dobra dla współpracy z klientem pod względem psychologicznym - regularna możliwość wglądu w coraz bardziej **działający** produkt daje mu poczucie pewności co do przyszłości projektu.

W Ruby on Rails na każdym kroku natykamy się na elementy zwinne. Tabele bazy danych są tworzone za pomocą migracji, za ich pomocą także można je prosto modyfikować. W ten sposób zamiast jednego wielkiego pliku z definicjami dziesiątek tabeli mamy kilkadziesiąt plików, dzięki którym możemy prześledzić wstecz historię rozrastania się bazy. Implikuje to w oczywisty sposób fakt, że można tworzyć jedynie niezbędne w danej chwili tabele, zaś późniejsze struktury i ich powiązania można zostawić na moment, w którym rzeczywiście będą potrzebne. Konwencje nazewnicze frameworka także wspierają projektowanie zwinne, gdyż w możliwie dużym stopniu niektóre funkje i komendy przypominają język naturalny, co ułatwia zrozumienie kod bez konieczności przeglądania dokumentacji.

5.1.5 Model-View-Controller

"Model-View-Controller zakłada podział aplikacji na trzy główne warstwy

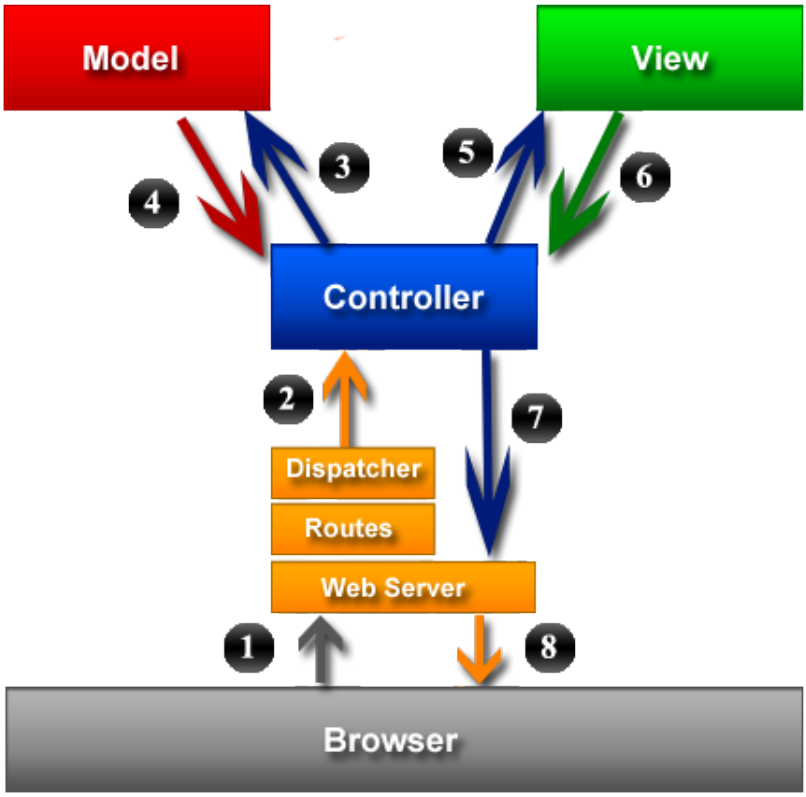
- **Model** - jest pewną reprezentacją problemu bądź logiki aplikacji.
- **Widok** - opisuje, jak wyświetlić pewną część modelu w ramach interfejsu użytkownika. Może składać się z podwidoków odpowiedzialnych za mniejsze części interfejsu.
- **Kontroler** - przyjmuje dane wejściowe od użytkownika i reaguje na jego poczynania, zarządzając aktualizacje modelu oraz odświeżenie widoków.

Wszystkie trzy części są ze sobą wzajemnie połączone"[5]

Zaprezentowana wyżej definicja została zaczerpnięta z Wikipedii. Choć jest to mocno skrócony opis przepływu informacji i generowania widoków za pomocą MVC daje ogólny pogląd na działanie tego schematu. Sercem aplikacji opartej o MVC są kontrolery - przetwarzają one zapytania pobierane z widoków, kierują je do odpowiednich funkcji i źródeł danych, zwracając przez widoki ostateczną odpowiedź dla użytkownika. Architektura ta została zaproponowana w roku 1979 przez norweskiego programistę Trygve Reenskauga, pracującego nad obiektowym językiem Smalltalk. Zaletą MVC jest bardzo wyraźny podział na części kodu zajmujące się obsługą baz danych (Model), fragmenty odpowiedzialne za wyświetlanie wiadomości (Widok) oraz na kontrolery, które zarządzają całością aplikacji. Warstwy nie powinny ingerować w swoje działanie. Ruby on Rails stosuje pasywną wersję architektury MVC - obiekty nie zmieniają samoczynnie swojego stanu. Aby nastąpiły zmiany musi nastąpić komunikacja z kontrolerami.

5.1.6 Struktura programu wobec architektury MVC

Ruby on Rails odgórnie narzuca architekturę MVC i choć oczywiście istnieją sposoby, aby korzystać z innych architektur, całe wsparcie frameworka najskuteczniej działa dla tego właśnie schematu. Ruby on Rails domyślnie stosuje zasadę, że dla każdego modelu istnieje jeden kontroler, natomiast widoki są ściśle powiązane z metodami kontrolera, dzieląc takie same nazwy. Pliki modeli i kontrolerów są pisane w Rubym, natomiast pliki widoków w specjalnych plikach html.erb, które są połączeniem pliku html z kodem Ruby umieszczanym między odpowiednimi znacznikami. Przy żądaniu wyświetlenia pliku html najpierw uruchamiany jest kontroler, który komunikując się z modelem zwraca do widoku potrzebne wartości. Dodatkowymi plikami są tzw. "helpery". Są to pliki powiązane z kontrolerami zawierające potrzebne metody, jednak nie mające swoich własnych plików z widokami.



5.1.7 Object-relational mapping

Mapowanie relacyjno-obiektowe jest rozwiązaniem stosowanym m.in. w Railsach przy obsłudze bazy danych, pozwalającym na natychmiastowe przyporządkowanie rekordów z bazy obiektom klas aplikacji. Pozwala to na wyraźne rozróżnienie między operacjami na bazie danych, a samą jej konfiguracją. Dzięki temu zmiana typu używanej bazy danych nie wnosi nic więcej jak modyfikacja pliku konfiguracyjnego - wszystkie zapytania do tabel cały czas powinny być obsługiwane. Warunkiem koniecznym i wystarczającym, aby w danej technologii można było używać ORM jest relacyjność bazy danych i obiektowość języka programowania. W Ruby biblioteką wspierającą mapowanie relacyjno-obiektowe jest ActiveRecords. Dla innych języków programowania istnieje także wiele rozwiązań wspierających ten model, takich

jak m.in. Hibernate(Java), Django(Python), Symfony(PHP).

5.2 Funkcjonalności

Jak zdecydowana większość programów sieciowych głównym odbiorcą aplikacji będzie zwyczajny użytkownik, wydzielona część z nich będzie mogła jednak wykonywać na stronie więcej czynności. Uprawnienia w aplikacji działają hierarchicznie, czyli administrator może zrobić wszystko to samo co zwyczajny użytkownik, ale będzie miał dodatkowe funkcje dostępne

Użytkownik

1. Założenie konta
2. Zalogowanie się na konto
3. Wylogowanie się z konta
4. Edycja danych konta
5. Wysyłanie i odbieranie prywatnych wiadomości od innych użytkowników
6. Komentowanie artykułów
7. Usuwanie własnych komentarzy
8. Przeglądanie listy turniejów, rund, partii
9. Przeglądanie partii zawodników
10. Dodawanie partii do listy obserwowanych oraz usuwanie z tej listy
11. Przeglądanie pojedynczej partii w oknie graficznym
12. Przeglądanie zapisu partii i nawigacja za jego pomocą do odpowiednich miejsc w rozgrywce
13. Przeglądanie odpowiednich wariantów dla obecnej pozycji
14. Oglądanie relacji na żywo dostarczonych z zewnętrznych źródeł danych
15. Dodawanie wiadomości dotyczących turniejów

Moderator treści

1. Usuwanie komentarzy innych użytkowników
2. Dodawania i usuwanie informacji z panelu komentatorskiego turnieju
3. Dodawanie i usuwanie artykułów

Moderator strony

1. Dodawanie nowych partii, rund i turniejów
2. Dodawanie nowych źródeł transmisji turniejów

Administrator

1. Usuwanie innych użytkowników
2. Zmiana uprawnień innych użytkowników

5.3 Pierwsza iteracja

W ramach pierwszej iteracji zostały zaimplementowane bazowe klasy dla całej aplikacji - moduł opisujący użytkownika oraz wspierający funkcję uwierzytelniania na stronie wraz z obsługą sesji. Powstała także grupa modułów opisujących partię szachową (klasy Tournament, Round, Game, Position, Player). Najważniejszą funkcjonalnością, która została dodana w trakcie tej iteracji był komponent wyświetlający partię szachową w formie graficznej. Niemal równie ważne było zaprogramowanie niewidocznych dla użytkownika funkcji parsujących pliki z partiami (w formacie PGN) i dodającym informacje do bazy danych. Metody przekształcające plik tekstowy w hierarchiczną strukturę bazy danych zostały stworzone przy pomocy biblioteki Chesspresso. Udostępnia ona funkcję, które odczytują z pliku zarówno informacje opisowe partii, jak i odczytujące i sprawdzające poprawność posunięć. Biblioteka ta, mimo wszystkich swoich zalet, miała także jedną wadę - jako iż została napisana w Javie, nie było łatwego sposobu umieszczenia domyślnej przeglądarki chespresso do kodu html. Chcąc uniknąć umieszczania apletów Javy zdecydowałem się na wprowadzenie kolejnej biblioteki - jChess. Przeglądarka graficzna partii po pierwszej iteracji była niemal identyczna jak domyślna szachownica stworzona przez komponent jChess. Wyświetlała ona partie, pozwalając za pomocą buttonów przesuwać partię do przodu i do tyłu. Zaimplementowane zostały możliwości wyświetlania informacji o turniejach, wynikach końcowych oraz rezultatach w poszczególnych rundach. Powstały także strony umożliwiające podgląd wyników poszczególnych zawodników.

Jako dodatek do pierwszej iteracji zostały dodane klasy i funkcji pozwalające użytkownikom na wysyłanie miedzy sobą wiadomości prywatnych, pisanie artykułów oraz dodawanie do nich komentarzy. Powstały też okna, w których użytkownicy mogą wyświetlać na żywo swoje opinie dotyczące całych turniejów, ale i poszczególnych partii.

5.4 Ewaluacja z użytkownikami po pierwszej iteracji

Po zakończeniu pierwszej iteracji nastąpiła faza ewaluacji ówczesnego stanu aplikacji. Badania przeprowadzałem głównie na szachistach, jednak brała w nich udział także osoba bez doświadczenia z korzystania z programów szachowych. Stosowałem zasadę, iż błędy były naprawiane w czasie między badaniami (o ile arbitralnie uznałem, że uwaga jest zasadna).

Dzięki tej zasadzie pozostałe osoby odkrywały nowe błędy, których naprawienie ulepszało jakość program. W wyniku badań zostało zauważonych kilkanaście błędów oraz zasugerowanych kilka alternatywnych rozwiązań. Najważniejsze z sugestii dotyczyły:

- Korzystania ze strzałek na klawiaturze w celu przesuwania figur;
- W oknie pokazywania partii szachowej brakowało linków do turniejów, do którgo należała partia i do rundy, w której była grana;
- Umieszczenia w zazwyczaj pustym miejscu na prawym panelu informacji o najnowszych artykułach;
- Ułatwienia użytkownikowi dostępu do szybkich linków przenoszących ich między powiązanymi stronami;
- Utworzenia stron, na których użytkownik może znaleźć albo szachistów, albo innych użytkowników;
- Wyróżnienia aktualnie przeglądanych na liście turniejów;
- Wyświetlania niektórych elementów interfejsu (głównie linków i napisów). Szczególnie często uwaga zwracana była na kontrast między kolorem napisów a kolorem tła;
- Błędów w działaniu strony: nie działających funkcji, błędnie wyświetlanych danych.

Na podstawie wyników ewaluacji z użytkownikami powstała lista elementów, które koniecznie trzeba naprawić, a także lista zmian, których wprowadzenie było opcjonalne. Wnioski z pierwszej listy zostały w pełni wykorzystane i błędy na niej zawarte zostały usunięte z aplikacji. W przypadku drugiej listy zostały wprowadzone wszystkie poprawki spełniające dwie cechy: brak kolizji z moją wizją programu oraz możliwość wprowadzenia bez gruntownego modyfikowania struktury programu.

5.5 Druga iteracja

Po wprowadzeniu zmian, które wynikły z ewaluacji z użytkownikiem nastąpił etap planowania kolejnego etapu pracy. Podstawową funkcją, jaka została zaplanowana na ten etap pracy było podłączenie działającego silnika szachowego do wygenerowania analiz pozycji. Sam silnik nie został samodzielnie zaimplementowany - wykorzystałem dostępny na licencji freeware o nazwie XXX.[51] Wyniki tych analiz miały się ukazywać przy przeglądaniu partii i dynamicznie zmieniać treść w zależności od wykonywanych posunięć. Dodatkową funkcjonalnością, na której rozwinięcie się zdecydowałem, było ułatwienie umieszczania okien pozwalających na pokazywanie transmisji partii z zewnętrznych źródeł danych. Z dodatkowych opcji uwzględniona została opcja dodawania komentarzy do poszczególnych pozycji każdej partii. Trzecią zmianą, jaką wniosła do projektu trzecia aplikacja, było zastosowanie okna Javy do wyświetlania czterech partii na raz. Zaimplementowane zostały także przycisku służące dynamicznemu zmienianiu wyświetlanych partii.

[51] Informacje o silniku YYY

5.6 Ewalucja po drugiej iteracji

5.7 Podsumowanie etapu tworzenia aplikacji

W wyniku dwóch iteracji powstała działająca i przetestowana aplikacja, realizująca wiele funkcji dostępnych zarówno od strony użytkownika, jak i tych działających w ukryciu dla późniejszych gości strony. Udało się zrealizować większość planów, jakie stawiałem przed aplikacją - dostępne są analizy komputerowe, można sprawnie nawigować partią szachową, przeglądać wyniki i informacje, komentować i porozumiewać się z innymi użytkownikami strony. Już teraz planuję dalsze iteracje, które będę wykonywał już po zamknięciu projektu magisterskiego - być może także będzie się w nich mieściła budowa biblioteki obsługującej partie szachowe. Po dwóch iteracjach aplikacja składa się z około 400 plików(przy czym dla większości szkielet został utworzony z użyciem railsowych generatorów kodu) oraz około 50 różnych klas. Graficznie program opiera się na arkuszach stylów wzbogaconego o rozwiązanie Blueprint. W trakcie rozwijania aplikacji korzystałem z następujących programów:

- E-texteditor - do pisania kodu w ruby, css, javascript i html.erb
- Eclipse - do przeglądania klas biblioteki Chesspresso
- GitHub - jako repozytorium[52]
- MySQL Workbench 5.0 OSS - do tworzenia schematu bazy danych
- Różne programy graficzne (m.in. Photoshop 12, GIMP)

W trakcie tworzenia aplikacji strona była sprawdzana na następujących przeglądarkach:

- Mozilla Firefox, wersje 3 i 4
- Google Chrome, wersja 11
- Opera, wersja 11.11
- Internet Explorer, wersja 8

Wygląd aplikacji różni się nieznacznie na każdej przeglądarce. Wynika to z drobnych rozbieżności w interpretowaniu plików CSS w tych przeglądarkach. Pod względem wydajności działania najskuteczniejszą przeglądarką okazała się Google Chrome, niewiele wolniejsza była zaś Opera. Najgorsze wyniki osiągała przeglądarka Microsoftu. Podczas pisania aplikacji nauczyłem się nie tylko nowego języka i jego frameworku do tworzenia stron www, ale spędziłem długie godziny zmagając się z integracją z istniejącymi komponentami napisanymi w innych językach. Koniec końców musiałem się także nauczyć modyfikować kod napisany przez innych aby dostosować go do mojej aplikacji.

6 Podsumowanie

Dziedzina o nazwie „Szachy i informatyka" jest niezwykle rozległa i bez większych trudności można napisać całe książki na temat rozmaitych algorytmów i ich wykorzystywania w rozwiązywaniu problemów szachowych. Ciekawym zagadnieniem są pytania nie mające praktycznego zastosowania w grze (takie jak słynne problemy ośmiu hetmanów lub tzw.

[52]https://github.com/PadawanBreslau/ZPionka

"problem skoczka"), jednak jeśli ograniczymy się do prawdziwych problemów to wciąż jest to fascynujące zagadnienie dla informatyków. Szachy, podobnie jak większość tego typu gier, są problemem PSPACE-zupełnym, z czego wynika, że nie jesteśmy w praktyczny sposób rozwiązać wielu problemów z nimi związanych. Programiści z całego świata walczą o duże pieniądze na wciąż rozwijającym się rynku programów szachowych. Rywalizują oni na rozmaitych polach: wśród komercyjnych baz danych, silników analizujących, książek debitowych czy też tablic końcówek. Następują także ciągłe zmagania na polu inżynierii programowania, budowy interfejsów czy udostępnianych funkcjonalności. Niektórzy programiści szukają w szachach prawd matematycznych, niektórzy próbują osiągać jedynie wyniki czysto praktyczne. Ważnymi kwestiami są dobór odpowiednich heurystyk, aproksymacja dla uzyskania praktycznego wyniku, obsługa sytuacji wyjątkowych czy tworzenie oddającej rzeczywistość funkcji liniowej do ewaluacji pozycji. W ostatnich latach zanotowano wielki rozwój praktycznych rozwiązań szachowych, teoria zaś nie zmienia się od lat. Możliwość rozstrzygnięcia, jaki wynik jest osiągany w przypadku idealnej gry obu stron, jest w dalszym ciągu zadaniem nierozstrzygalnym przy obecnej architekturze systemów komputerowych.

7 Bibligrafia

Literatura

[1] Dave Thomas, *Agile. Programowanie w Rails.*, Wydawnictwo Helion 2008, Tłumaczenie: Krzysztof Szafranek

[2] Claude E. Shannon *Programming a Computer for Playing Chess1*, Philosophical Magazine, 1950

[3] John Tromp *Number of chess diagrams and positions*, John's Chess Playground, http://homepages.cwi.nl/ tromp/, 2010

[4] prof. Andrzej Kisielewicz *Sztuczna inteligencja i logika*, Wydawnictwo Naukowo-Techniczne, 2011

[5] *http://pl.wikipedia.org/wiki/Model-View-Controller*