Domain Event Driven pada Aplikasi Komputer Catur: Logika Catur

Tati Suprapti[1], Ryan H. Silalahi[2,#]

Abstrak— Saat ini komputer catur telah berkembang sedemikian rupa sehingga banyak elemen yang ada pada aplikasi komputer catur dapat dipisahkan dan dipasang pada aplikasi lain. Sebagai contoh adanya *engine* catur yang berfokus hanya untuk penerapan kecerdasan buatan. Adapula GUI yang hanya berfokus pada papan dan buah catur tanpa adanya aturan sehingga dapat dijadikan permainan lain dengan tampilan "catur". Dengan menggunakan sudut padangan *domain event driven* maka dapat kita cermati bahwa elemen-elemen pada sebuah aplikasi dapat dipisahkan sesuai dengan domainnya. Tulisan ini dibuat untuk mendefinisikan domain aturan catur sebagai salah satu elemen yang dapat digunakan pada saat pengembangan aplikasi komputer catur. Dengan berfokus pada aturan catur sebagai domainnya kita dapat membuat sebuah pustaka (*library)* aplikasi untuk dijadikan referensi oleh elemen-elemen lain pembentuk aplikasi. Adapun *library* aturan catur ini dapat dinamakan *chess logic.*

Kata kunci— Komputer catur, aturan catur, logika catur, domain

Abstract— Currently the chess computer has evolved in such way that many of the elements that exist in the application of computer chess can be separated and placed in another application. As an example of the chess engine that focuses only on the application's artificial intelligence. There is also a GUI that focus solely on board and chess pieces in the absence of rules so that it can be used as another application as chess interface. By using domain event driven point of views, we can look at that elements in an application can be separated according to their domain. It's intended to define the domain rules of chess as one of the elements that can be used during the development of computer chess application. By focusing on the rules of chess as a domain we can create a library (library) application to be used as a reference by other application's elements forming an application. The library's rules of chess can be called chess logic.

Keywords— Chess computer, chess rule, chess logic, domain

I. PENDAHULUAN

Aturan catur saat ini sudah didefinisikan oleh FIDE sebagai organisasi catur dunia. Oleh karena itu akan dengan mudah kita pahami bahwa umumnya permainan catur merujuk pada aturan yang ditulis oleh FIDE.

Salah satu elemen yang sering dipisahkan pada aplikasi komputer catur ialah elemen *AI* (*engine*) catur. Dan saat ini sudah banyak kejuaraan yang mempertandingkan antar mesin sehingga dapat diurut tingkat kekuatan dari *AI* yang dipertandingan. Dengan adanya konsep dipisahkannya *AI* dengan aplikasi utama, maka bermunculan pula konsep elemen lain dari aplikasi komputer catur yang dapat dipisahkan. Salah satunya ialah aturan catur itu sendiri.

Pada sebuah aplikasi komputer catur atau pada permainan lainnya tentu ada sebuah aturan yang digunakan pada perminan tersebut. Hal ini merupakan sebuah domain pasti yang ada pada setiap permainan.

Sebuah domain aturan catur sangat perlu setidaknya membahas tentang status siapa yang melangkah, apakah permainan telah selesai atupun masih berlangsung dan langkah manasaja yang dapat dilakukan (*legal moves).*

Pada prakteknya setiap domain aturan catur mempunyai sebuah reprenstasi papan sehingga dapat diketahui posisi setiap buah catur pada papan, metode representasi papan sudah bukan metode yang baru, pengembang dapat menggunakan salah satu metode yang telah ada, salah satunya ialah metode 0x88.

Oleh karena itu tulisan ini mencoba mendefiniskan domain aturan catur sehingga dapat dijadikan sebuah referensi dari *library* aturan catur atau yang dapat kita namakan *chess logic.*

II. KEGUNAAN DAN PENEMPATAN

Kegunaan atau penempatan *library chess logic* dapat digunakaan pada banyak hal dalam aplikasi komputer catur. Berikut contoh penempatannya secara umum:

A. Aturan pada GUI

Beberapa *library GUI* catur seperti *chessboard.js* berfokus hanya pada tampilan dari papan dan buah catur, maka dari itu fungsi *chess logic*, berperan sangat penting untuk membangun sebuah *des*ign interaksi yang berdasarkan aturan catur.

B. Validasi Langkah dari Pemain

Khusus pada aplikasi yang membuat tampilan dan pemrosesan terpisah seperti arsitektur *website*, sangat diperlukan validasi terhadap setiap apa data yang dimasukan oleh pengguna/pemain. *Chess logic* akan sangat berguna untuk memvalidasi data seperti langkah yang masuk hanya boleh langkah sah.

C. Pengawasan Status Permainan

Seiring dengan permainan yang dijalankan, maka kondisi berakhir pemainan harus dapat dideteksi untuk menghentikan permainan. Seperti kondisi 3 posisi sama atau 50 langkah tanpa pemukulan haruslah dapat diketahui. Hal ini tentu dapat menggunakan *Chess logic* sebagai pengecekan status permainan.

D. Validasi Permainan

Seringkali dalam sebuah aplikasi komputer catur terdapat fitur *save/load.* Fitur ini dapat menggunakan format PGN sebagai *file* data permainannya. Tetapi apabila permainan dipanggil kembali. Aplikasi haruslah dapat memvalidasi data

Artikel diterima 19 Desember 2016; direvisi 18 Januari 2017; disetujui 20 Januari 2017; dipublikasikan Februari 2017

[1]Program Studi Teknik Informatika, STMIK AMIK Bandung, Jl. Jakarta No. 28, Bandung, Jawa Barat 40272, Indonesia
[2]Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, Jl. Ganesha No. 10, Bandung, Jawa Barat 40132, Indonesia
[#] E-mail: mr.ryansilalahi@gmail.com

permainan sesuai dengan aturan sehingga tidak ada kesalahan selama pemanggilan kembali (*load*) permainan. Dalam hal ini *Chess logic* yang mengimplementasikan fitur validasi format permainan seperti PGN dapat sangat berguna untuk membuat data permainan atau validasinya.

III. Representasi Papan

Tahap awal untuk membuat sebuah *chess logic* ialah dengan cara merepresentasikan papan dan buah catur itu sendiri. Banyak metode yang telah ada seperti 0x88, *bitboards*, *vector-attacks*, dan masih banyak lagi. Bahkan dengan cara membuat *array* 2 dimensi yang masing-masing sebanyak 8 kolom dan 8 baris dapat gunakan untuk representasi papannya.

Representasi papan yang dimaksud dalam *library* tentunya bukan representasi secara visual, hanya berupa representasi secara abstraksi untuk dapat digunakan oleh komputer.

Setelah mengimplementasikan representasi papan, maka dapat dimulai dengan mencoba posisi-posisi buah catur untuk pengujiannya. Salah satu cara cara pengujian yang mudah ialah dengan membuat sebuah fitur yang mencetak papan dalam bentuk *string* agar dapat ditampilkan dalam *console*.

Fitur dari *chess logic* yang sangat berkaitan dengan representasi papan ialah fitur penempaan buah catur (*piece placement*).

Berikut contoh abstraksi API dari fitur penempatan buah catur dan *debuging* posisi papan:

```
// piece placement
bool put(String square, String piece);
bool get(String square);

// debuging string or array
String printBoard();
Array printBoard();
```

IV. Aturan Langkah

Aturan langkah pada intinya ialah langkah yang masuk dalam kategori langkah sah dan status permainan masih dalam posisi berjalan. Sebagai media komunikasi langkahnya ada 2 jenis format langkah yang dapat kita gunakan dalam *library chess logic*, yaitu:

A. *SAN (Standard Algebraic Notation)*

Format langkah menggunakan *SAN* merupakan salah satu format langkah yang paling umum digunakan. Contoh bentuk langkah nya seperti e4, e5, Nf3, Nc6 dapat dibaca oleh banyak pemain catur yang mengerti notasi catur. Oleh karena itu format ini dapat digunakan pada *internal* aplikasi tetapi dapat dibaca oleh manusia.

B. *Petak ke Petak*

Format ini merupakan format yang mudah untuk digunakan dikarenakan dapat diketahui dengan jelas buah catur di petak mana yang melakukan langkah, dan petak tujiannya. Berikut contoh langkahnya: e2e4, e7e5, g1f3, b8c6. Dengan melihat pada 2 huruf pertama sebagai petak asal dan 2 huruf terakhir sebagai petak tujuan.

Hal yang perlu diperhatikan apabila hendak menggunakan format ini ialah ketika bidak mencapai baris akhir atau dinamakan promosi. Perlu adanya parameter tambahan untuk keterangan promosinya, bisa menggunakan parameter tambahan atau penambahaan format seperti e7e8=Q.

Berikut contoh abstraksi API dari fitur langkah:

```
// move
bool moveSAN(String san);
bool moveSquare(String s2s);
```

V. Langkah Sah

Sebagai salah satu peran penting *library chess logic* ialah untuk mengetahui langkah sah yang boleh dilakukan. Seperti dalam GUI catur yang belum mempunyai aturan catur, langkah yang diperbolehkan haruslah berdasarkan ketentuan langkah yang sah. Fitur ini merupakan salah satu fitur utama dari *library chess logic*.

Berikut contoh abstraksi API dari fitur langkah sah:

```
// legal moves
Array legalMoves();
```

VI. Status Permainan

Status permainan paling dasar yang perlu diketahui ialah status permainan masih berjalan, salah satu pemain menang dengan *checkmate*, atau status permainan imbang seperti konsisi 3 langkah sama, *stalemate*, kekurangan materi, dan 50 langkah tanpa pemukulan.

Berikut contoh abstraksi API dari status permainan:

```
// status
bool isGameOver();
bool isCheckmate();
bool isWhiteWinner();
bool isBlackWinner();
bool isDraw();
bool isStalemate();
bool isInsufficientMaterial();

// langkah tanpa pemukulan
bool isHalfMovesExceeded();
```

VII. Pemanggilan/Penyimpanan/Jejak Permainan

Fitur ini merupakan fitu yang bersifat opsional, tetapi bisa menjadi sangat penting seperti pada beberapa aplikasi yang dapat mendukung fitur *save/load*.

Format data penyimpanan yang sering digunakan saat ini ialah FEN untuk posisi permainan, dan PGN untuk data lengkap permainan. Tentu haruslah mendukung untuk validasi, generasi, maupun pemanggilan (*load*) data FEN/PGN.

Berikut contoh abstraksi API dari manipulasi data/jejak permainan:

```
// FEN
bool validateFEN(String fen);
bool loadFEN(String fen);
bool generateFEN();

// PGN
bool validatePGN(String pgn);
bool loadPGN(String pgn);
bool generatePGN();
```

VIII. PENGUJIAN

Pengujian tentang akurasi dari aplikasi komputer catur salah satu caranya dapat menggunakan metode *move generation*, dalam hal ini *perft* sebagai salah satu metode yang banyak dipakai dapat digunakan untuk melihat jumlah langkah sah dalam kedalaman tertentu.

Dengan merujuk pada hasil pengujian menggunakan *perft*, maka dapat dilihat akurasi dari *library chess logic* yang dibuat. Dapat pula dilihat tingkat kecepatan pemrosesan *perft* sebagai acuan untuk melihat seberapa efisien kode yang dibuat.

IX. KESIMPULAN DAN SARAN PENGEMBANGAN

Pemanfaatan konsep *domain event driven* dapat memberikan sudut pandang yang baru untuk inovasi pada rekayasa perangkat lunak. Salah satunya pada aplikasi komputer catur.

Dengan membuat sebuah *library chess logic* yang memfokuskan pada aturan catur, dapat dimanfaatkan dibanyak elemen/aspek pada saat pembuatan aplikasi komputer catur.

Dalam pengembangan *library chess logic* tahap pertama ialah untuk implementasi fitur: penempatan buah catur, langkah, langkah sah, dan status permainan. Adapun tahap selanjutnya bila diperlukan ialah fitur data/jejak permainan dengan format FEN/PGN.

REFERENSI

[1] FIDE. (2009, Juli). FIDE LAWS of CHESS. FIDE . Dresden, Jerman. Pdf. Available: https://www.fide.com/FIDE/handbook/LawsOfChess.pdf.

[2] What is Domain-Driven Design?, http://dddcommunity.org/, 2016.

[3] UCI, https://chessprogramming.wikispaces.com/UCI, 2016.

[4] Board Representation, https://chessprogramming.wikispaces.com/Board+Representation, 2016.

[5] Algebraic Chess Notation, https://chessprogramming.wikispaces.com/Algebraic+Chess+Notation, 2016.

[6] Forsyth-Edwards Notation, https://chessprogramming.wikispaces.com/Forsyth-Edwards+Notation, 2016.

[7] Portable Game Notation, https://chessprogramming.wikispaces.com/Portable+Game+Notation, 2016.

[8] Perft Results, https://chessprogramming.wikispaces.com/Perft+Results, 2016