Jurnal Ilmiah Universitas Satya Negara Indonesia

Published by : LP3M Universitas Satya Negara Indonesia
Vol. 02 No.02, 2024
Page 48-57
P-ISSN : 1979-5246

IMPLEMENTATION OF BINARY SEARCH TREE METHOD FOR ANDROID-BASED MORSE CODE DESCRIPTION

PENERAPAN METODE *BINARY SEARCH TREE* UNTUK DESKRIPSI SANDI MORSE BERBASIS ANDROID

Nurul Chafid[1*], Apriyanto Saputra [2]
[1*,2] Universitas Satya Negara Indonesia, Jakarta Selatan
[1]chafidaja@usni.ac.id, [2]student810.usni@gmail.com

INFORMASI ARTIKEL
Submitted:
10-01-2024

Accepted:
21-02-2024

Published:
22-05-2024

Keywords:
Binary Search Tree, Android, Morse code
Kata Kunci:
Binary Search Tree; Android; Sandi Morse

ABSTRACT

Morse code is a code that is still used by ORARI and the Scout in communicating. Learn Morse code requires an understanding of the cues delivered because Morse code was not submitted as a visualization of dots and lines, but must understand the long and short signal of Morse code. Morse tree method is used to facilitate the deciphering Morse code message. The aim of this study is to apply the method in a binary search tree and implements Morse on android platform in the form of short messages using Morse code. Methods of approach and the development of systems using object-oriented approach and waterfall method. Results from this study that to change or alter the text to Morse be required text identification numbers used to trace the Morse tree with a binary search tree method. The application of Morse code signaling carried out in the interval of 200 milliseconds for every point. Binary tree search method can be applied in tracing the Morse tree and implemented on the android platform for deploying long and short signal Morse code.

ABSTRAK

Sandi morse merupakan sandi yang masih digunakan oleh ORARI dan Gerakan Pramuka dalam berkomunikasi. Mempelajari sandi morse membutuhkan pemahaman terhadap isyarat yang disampaikan karena sandi morse tidak disampaikan sebagai visualisasi titik dan garis namun harus mengerti isyarat panjang dan pendek sandi morse. Metode pohon morse digunakan untuk mempermudah dalam mengartikan pesan sandi morse. Tujuan dari penelitian ini adalah menerapkan metode *binary search tree* dalam menelusuri pohon morse dan mengimplementasikannya pada *platform* android berupa pengiriman pesan singkat menggunakan sandi morse. Metode pendekatan dan pengembangan sistem menggunakan meode pendekatan berorientasi objek dan metode *waterfall*. Hasil dari penelitian ini bahwa untuk mengubah teks menjadi morse atau mengubah morse menjadi teks diperlukan nomor identitas yang digunakan untuk menelusuri pohon morse dengan metode *binary search tree*. Adapun pengaplikasian isyarat sandi morse dilakukan dalam interval selama 200 milidetik untuk setiap titiknya. Metode *binary search tree* bisa diterapkan dalam menelusuri pohon morse dan diimplementasikan pada *platform* android untuk mengaplikasikan isyarat panjang dan pendek sandi morse.

INTRODUKSI

Sandi morse adalah sandi yang tersusun atas titik (*dot*) dan garis (*dash*) yang mewakili huruf morse. Sandi morse merupakan salah satu sandi yang masih aktif digunakan antara lain dalam komunikasi CW (Continuous Wave) yang dilakukan oleh ORARI (Organisasi Amatir Radio Indonesia) dan pelatihan yang dilakukan oleh Gerakan Pramuka agar mampu mengirim dan menerima berita dengan sandi morse. Sandi morse digunakan sebagai alternatif pengiriman pesan dikarenakan keterbatasan peralatan komunikasi dalam kondisi berbahaya. Mempelajari sandi morse adalah keharusan bagi mereka yang berkecimpung dalam ORARI maupun Gerakan Pramuka. Mereka harus membedakan isyarat panjang dan isyarat pendek sandi morse karena sandi morse tidak hanya sebagai visualisasi titik dan garis saja (ORDA DKI BID ORG, 2005). Isyarat panjang adalah definisi dari simbol garis dan isyarat pendek adalah definisi dari simbol titik pada sandi morse.

Dalam buku saku pramuka terdapat metode pengelompokan dan metode pohon morse untuk mempermudah mempelajari sandi morse (Tim Pembina Pramuka, 2009). Metode pengelompokan yaitu mengelompokan huruf morse berdasarkan susunan titik saja, garis saja dan kombinasi keduanya. Pada Pohon morse memperlihatkan struktur sandi morse berdasarkan urutan titik atau garis yang membentuk pohon dengan dua cabang (pohon biner). Namun pada buku saku sandi morse diperlihatkan dalam bentuk teks saja tanpa mengetahui isyarat panjang dan pendeknya.

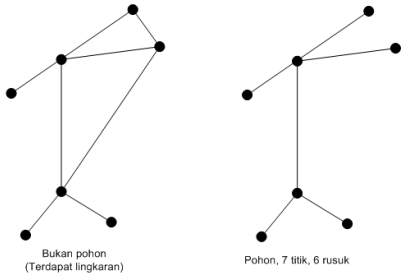
Metode *binary search tree* merupakan metode pencarian pada pohon biner (*binary tree*) dengan membandingkan nilai pencarian dengan setiap simpul pohon. Metode ini dapat digunakan untuk mengubah morse menjadi teks (Honggo, 2009). Namun mendeskripsikan sandi morse tidak hanya mengubah morse menjadi teks, mengubah teks menjadi morse juga harus bisa dilakukan.

Perangkat *mobile* sangat diminati masyarakat khususnya android. Pengguna android di Indonesia mencapai 58,73 % yang merupakan angka tertinggi sampai akhir Mei 2015 (Stat Counter, 2015). Penerapan metode *binary search tree* dalam suatu perangkat mobile membutuhkan bahasa pemrograman yang mendukung struktur data *tree* sehingga penerapan metode ini untuk deskripsi sandi morse dapat diimplementasikan pada perangkat *mobile*. Namun dengan adanya perangkat *mobile*, pendeskripsian sandi morse bisa dilakukan dengan memperlihatkat isyarat panjang dan pendek sehingga pengguna dapat berkomunikasi menggunakan sandi morse sekaligus sebagai media pembelajaran sandi morse.

Berdasarkan uraian latar belakang penelitian ini, terdapat rumusan masalah antara lain; Bagaimana menelusuri pohon morse untuk mendeskripsikan sandi morse, Bagaimana mengimplementasikan penelusuran pohon morse dalam suatu perangkat *mobile*, Bagaimana mendeskripsikan sandi morse khususnya isyarat panjang dan pendek dalam suatu perangkat *mobile*.

Pohon merupakan bentuk khusus dari *graph*. Menurut Wibisono (2013), pohon adalah sebuah *graph* yang mempunyai n buah simpul (titik), n-1 busur (rusuk) dan tidak mempunyai lingkaran (*cycle tree*) atau sirkuit serta merupakan *graph* terhubung.

Gambar 1. Pohon dan bukan Pohon

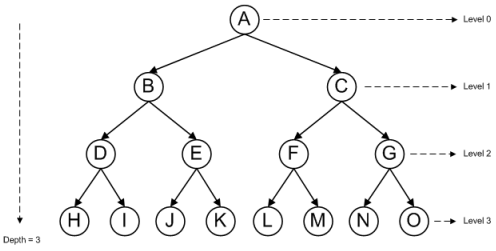


Menurut Wibisono (2013), pohon biner merupakan pohon dimana pada setiap titik cabang pohon dapat mempunyai dua anak cabang (satu sabang kiri dan satu cabang kanan), satu anak cabang, atau tidak mempunyai anak cabang.

Menurut Sjukani (2012), *strictly binary tree* adalah pohon biner yang semua simpulnya (kecuali simpul daun) mempunyai lengkap simpul sub-ordinat kiri dan sub-ordinat kanan.

Menurut Sjukani (2012), *complete binary tree* adalah pohon biner dengan kedalaman tertentu yang merupakan *strictly binary tree* dimana semua daun berada pada level yang sama.

Gambar 2. *Complete binary tree*



Adapun rumus untuk menentukan jumlah simpul pada *complete binary tree* yaitu

$$n = 2^{h+1} - 1$$

Keterangan:

n : jumlah simpul

h : level

Pohon pencarian biner memungkinkan pencarian data dapat dilakukan dengan cepat dan efisien. Data pada pohon ini bersifat unik dan tidak unik. Terdapat ketentuan dalam peletakan data dalam pohon biner agar pencarian dapat dilakukan dengan lebih mudah, yaitu dengan membandingkan infonya.

Jika x adalah info, maka berlaku ketentuan bahwa jika x memiliki subordinat kiri, nilainya harus lebih kecil dari x dan sebaliknya jika x memiliki subordinat kanan, nilainya harus lebih besar dari x.

Pohon AVL salah satu jenis *self balancing binary search tree* yang ditemukan oleh G.M. Adelson -Velskii dan E.M. Landis pada tahun 1962. Nama AVL merupakan gabungan singkatan kedua nama mereka. Menurut Sjukani (2012) untuk mengetahui sebuah simpul pohon menjadi tetap seimbang atau tidak, setiap simpul akan dianalisa dengan menggunakan kode sebagai berkut:

a. Nilai 0, berarti kedalaman cabang kiri sama dengan kedalaman cabang kanan
b. Nilai 1, berarti cabang pohon kiri lebih dalam dari cabang pohon kanan atau disebut berat ke kiri.
c. Nilai -1, berarti cabang pohon kanan lebih dalam dari cabang pohon kiri atau disebut berat ke kanan.

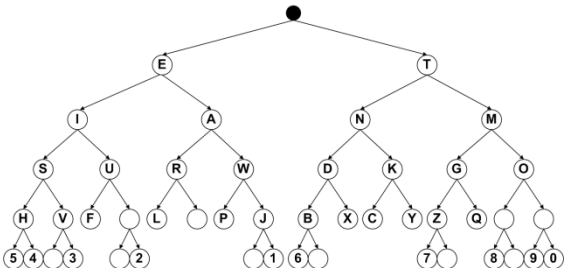
Sandi morse adalah sandi yang tersusun atas titik (*dot*) dan garis (*dash*) yang mewakili huruf alfabet, angka, tanda baca maupun tanda isyarat tertentu. Sandi morse bisa disampaikan dalam bentuk tulisan, bunyi, gerakan, dan sinar lampu.

Sandi morse diciptakan oleh Samuel F. B. Morse dan Alfred Vail pada tahun 1835.Dalam sandi morse internasional (Wikipedia, 2015), untuk menentukan isyarat panjang dan pendek sandi morse diatur dalam dua keadaan (*on* atau *off*) dengan ketentuan sebagai berikut:

a. Isyarat pendek, titik yaitu 1
b. Isyarat panjang, garis yaitu 111
c. Pemisah antara titik dan garis yaitu 0
d. Pemisah antara huruf satu dengan yang lain yaitu 000
e. Pemisah antara kata satu dengan yang lain yaitu 0000000

Metode yang digunakan dalam mempelajari sandi morse adalah metode pohon morse (Tim Pembina Pramuka, 2009) seperti gambar dibawah ini:

Gambar 3. Pohon morse



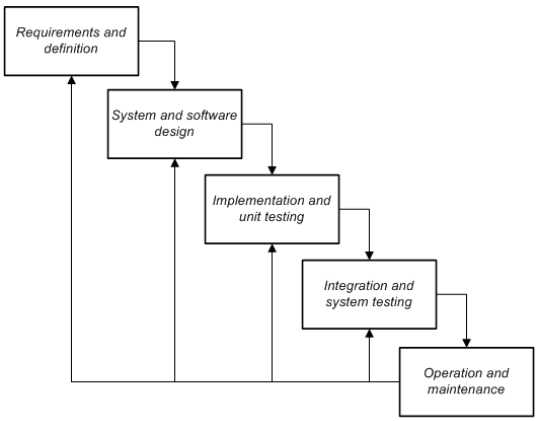
METODE PENELITIAN

Adapun desain penelitian yang dilakukan penulis adalah *research and development*. Penelitian ini difahami sebagai kegiatan penelitian yang dimulai dengan *research* dan diteruskan dengan *development*.

Penulis menggunakan data sekunder dan metode pengumpulan data yang digunakan adalah metode studi pustaka yaitu mencari referensi dari penelitian yang sudah pernah dilakukan, referensi dari buku-buku yang berhubungan dengan materi, dan referensi dari internet dengan membuka *website* sebagai pelengkap dalam penulisan skripsi ini.

Metode pendekatan yang digunakan yaitu metode pendekatan berorientasi objek. Metode pengembangan perangkat lunak yang dilakukan menggunakan model *Waterfall* yang terdiri dari (Sommerville, 2011:30)

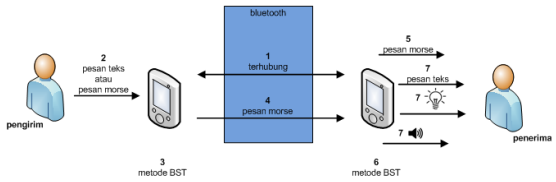
Gambar 4. Model *waterfall*



HASIL DAN PEMBAHASAN

Seperti yang sudah dibahas dalam penelitian sebelumnya sebagai penulis mencoba menerapkan metode algoritma tersebut dengan mengarahkan pada setiap *event* secara proses berututan dengan beberapa langkah proses pada *binary tree* berikut adalah gambaran proses kerja sistem saat pengirim mengirim pesan morse sampai pesan itu diterima oleh penerima dalam bentuk pesan morse dan diubah menjai pesan teks, isyarat bunyi dan isyarat sinar lampu oleh penerima pesan.

Gambar 5. Proses kerja system



Mengubah morse menjadi teks

Langkah 1 : Baca setiap simbol morse
Langkah 2 : Jika pembacaan mencapai urutan akhir maka menuju langkah ke-7 jika tidak menuju langkah ke-3
Langkah 3 : Telusuri dari akar dan menuju langkah ke-4
Langkah 4 : Jika karakter = vertical bar (|) maka baca simpul terakhir yang dikunjungi dan ulangi langkah penelusuran dari langkah ke-1 jika tidak menuju langkah ke-5
Langkah 5 : Jika karakter = titik (.) maka telusuri melalui cabang kiri dan ulangi langkah penelusuran dari langkah ke-1 jika tidak menuju langkah ke-6
Langkah 6 : Apabila karakter = garis (-) maka telusuri melaui cabang kanan dan ulangi langkah penelusuran dari langkah ke-1
Langkah 7 : Baca simpul

Mengubah teks menjadi morse

Langkah 1 : Baca setiap karakter dari teks
Langkah 2 : Jika pembacaan mencapai urutan akhir maka menuju langkah ke-9 jika tidak menuju langkah ke-3
Langkah 3 : Cari posisi karakter pada pohon morse dan menuju langkah ke-4
Langkah 4 : Jika posisi karakter ditemukan maka menuju langkah ke-5 jika tidak ulangi langkah penelusuran dari langkah ke-1
Langkah 5 : Telusuri dari akar dan menuju langkah ke-6
Langkah 6 : Jika penelusuran mencapai posisi yang dicari maka beri simbol vertical bar (|) dan ulangi langkah penelusuran dari langkah ke-1 jika tidak menuju langkah ke-7
Langkah 7 : Jika penelusuran melalui cabang kiri maka beri simbol titik dan ulangi langkah penelusuran dari langkah ke-6
Langkah 8 : Jika penelusuran melalui cabang kanan maka tandai beri simbol garis dan ulangi langkah penelusuran dari langkah ke-6
Langkah 9 : Baca simbol

Penyisipan

Langkah 1 : Buat simpul pertama dan beri kode = 0
Langkah 2 : Tempatkan simpul pertama sebagai simpul akar
Langkah 3 : Sisipkan simpul baru. Buat simpul baru dan beri kode = 0
Langkah 4 : Jika simpul baru lebih kecil dari simpul sekarang maka tempatkan di cabang kiri dan beri kode simpul sekarang = 1, jika simpul baru lebih besar dari simpul sekarang maka tempatkan di cabang kanan dan beri kode simpul sekarang = -1
Langkah 5 : Jika jumlah kode = 2 dari simpul baru maka putar ke kanan (right rotation), jika jumlah kode = -2 dari simpul baru maka putar ke kiri (left rotation).
Langkah 6 : Sisipakan simpul baru kembali

Pohon AVL untuk menyeimbangkan pohon biner.

Right rotation

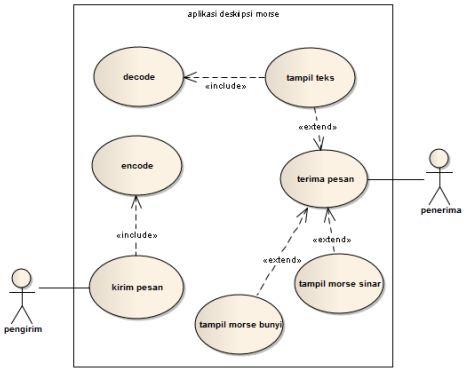
Langkah 1 : Posisi Q berada di cabang kiri AKAR
Langkah 2 : Posisi cabang kiri AKAR berada di cabang kanan Q
Langkah 3 : Posisi cabang kanan Q berada di AKAR
Langkah 4 : Posisi AKAR berada di Q
Langkah 5 : Beri kode simpul AKAR = 0 dan beri kode simpul cabang kanan AKAR = 0

Left rotation

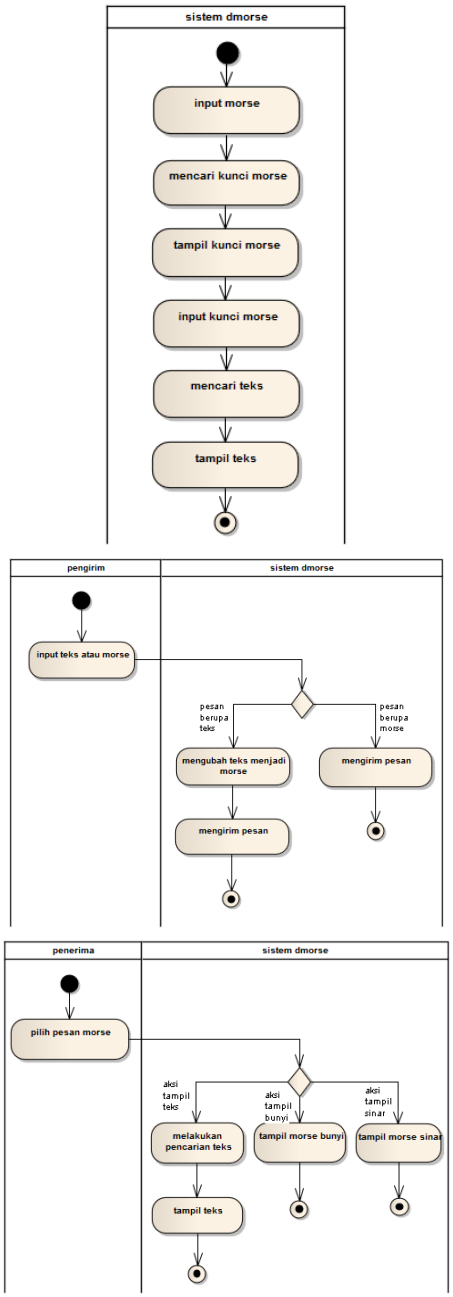
Langkah 1 : Posisi Q berada di cabang kanan Akar
Langkah 2 : Posisi cabang kanan AKAR berada di cabang kiri Q
Langkah 3 : Posisi cabang kiri Q berada di AKAR
Langkah 4 : Posisi AKAR berada di Q
Langkah 5 : Beri kode simpul AKAR = 0 dan beri kode simpul cabang kiri AKAR = 0

1. Diagram *Use Case*

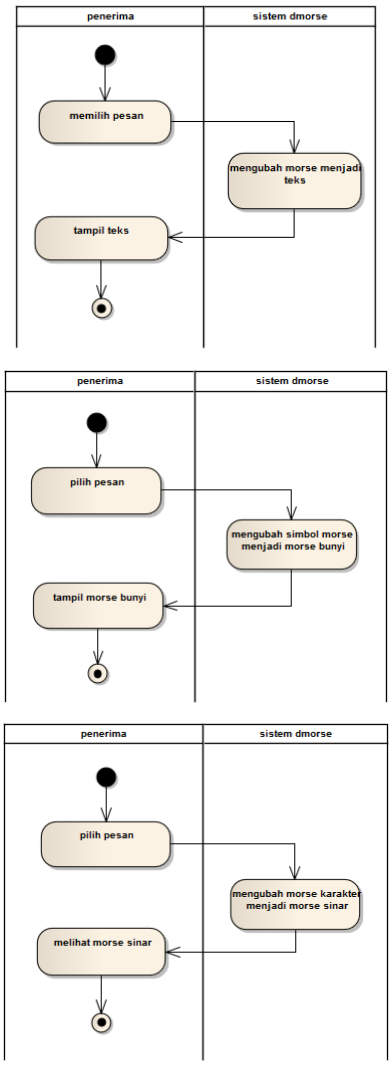
Gambar 6. Diagram *use case*



Gambar 7. Diagram *activity encode, activity decode,* dan *activity* kirim pesan

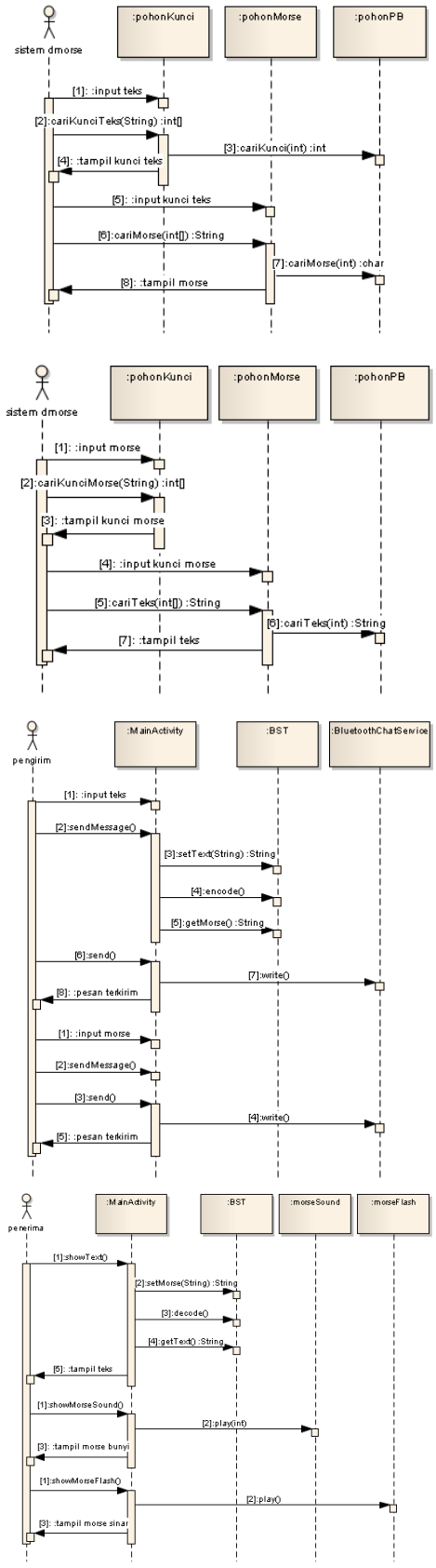


Gambar 8. Diagram *activity* morse teks, morse bunyi, morse sinar

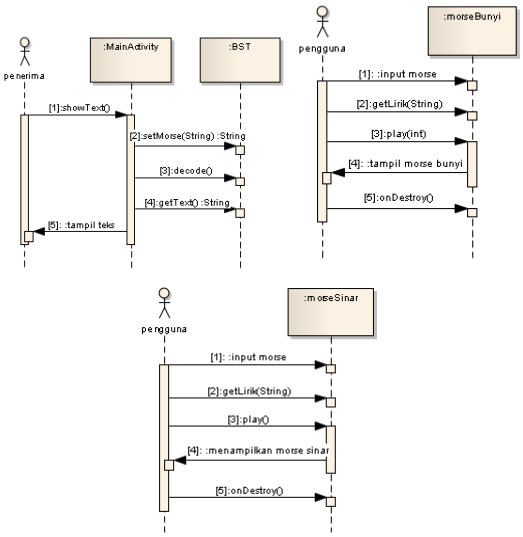


2. Diagram Sequence

Gambar 9. Diagram *sequence encode, decode,* kirim pesan dan terima pesan

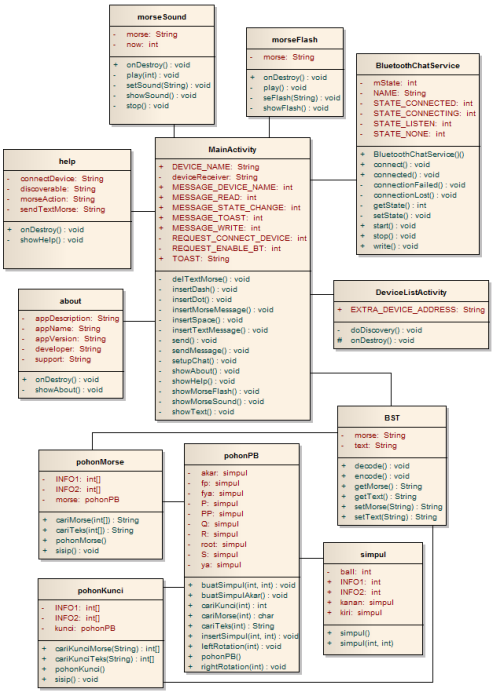


Gambar 10. Diagram *sequence* morse teks, morse bunyi, dan morse sinar



3. Diagram Class

Gambar 11. Diagram *class*



1. Mengubah teks menjadi morse

Tabel 1. Perubahan teks, kunci, dan morse

Teks	Dec ASCII Teks	Kunci	Dec ASCII Morse	Morse
T	84	48	45, 124	- |
O	79	60	45, 45, 45, 124	--- |
L	76	18	46, 45, 46, 46, 124	.-.. |
O	79	60	45, 45, 45, 124	--- |
N	78	40	45, 46, 124	-. |
G	71	42	45, 45, 46	--.

2. Mengubah morse menjadi teks

Tabel 2. Perubahan teks, kunci, dan morse

Morse	Dec ASCII Morse	Kunci	Dec ASCII Teks	Teks
-	45	64	84	T
|	124	32		
-	45	64	79	O
-	45	64		
-	45	64		
|	124	32		
.	46	0	76	L
-	45	64		
.	46	0		
.	46	0		
|	124	32		
-	45	64	79	O
-	45	64		
-	45	64		
|	124	32		
-	45	64	78	N
.	46	0		
|	124	32		
-	45	64	71	G
-	45	64		
.	46	0		

3. Menampilkan morse dalam isyarat bunyi dan sinar lampu

Tabel 3. Interval morse dan pemisah

Simbol Morse	Interval Morse (milidetik)	Interval pemisah (milidetik)	Jumlah (milidetik)
-	200 x 3	200 x 1	800
|	200 x 3		600
-	200 x 3	200 x 1	800
-	200 x 3	200 x 1	800
-	200 x 3	200 x 1	800
|	200 x 3		600

.	200 x 1	200 x 1	400
-	200 x 3	200 x 1	800
.	200 x 1	200 x 1	400
.	200 x 1	200 x 1	400
|	200 x 3		600
-	200 x 3	200 x 1	800
-	200 x 3	200 x 1	800
-	200 x 3	200 x 1	800
|	200 x 3		600
-	200 x 3	200 x 1	800
.	200 x 1	200 x 1	400
|	200 x 3		600
-	200 x 3	200 x 1	800
-	200 x 3	200 x 1	800
.	200 x 1	200 x 1	400

4. Implementasi perangkat keras

Tabel 4. Spesifikasi perangkat keras

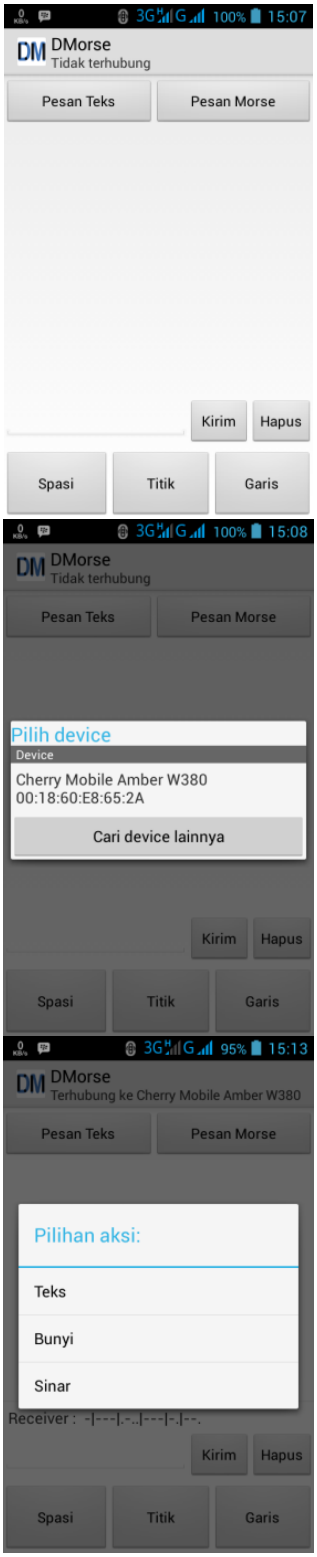
No	Nama Perangkat	Spesifikasi
1.	Processor	ARM 1.3 GHz
2.	Layar	480 x 800 pixel
3.	Memori	RAM 512 MB
4.	Storage	4 GB
5.	Speaker	Speaker perangkat mobile

5. Implementasi perangkat lunak

Tabel 5. Spesifikasi perangkat lunak

No	Nama Perangkat	Spesifikasi
1.	Sistem Operasi	Android 4.2.2 Jelly Bean
2.	API	Versi 17
3.	Java VM	Dalvik 1.6.0
4.	OpenGL ES	2.0

6. Implementasi antarmuka



Gambar 12. *User Interface*

KESIMPULAN

Adapun kesimpulan dari penerapan metode *binary search tree* untuk deskripsi sandi morse adalah sebagai berikut:

a. Pohon morse adalah salah satu metode untuk mempemudah mempelajari sandi morse. Dengan adanya pohon morse, metode *binary search tree* dapat diterapkan untuk mengubah sandi morse menjadi teks dan mengubah teks menjadi sandi morse berdasarkan kunci yang digunakan untuk menelusuri pohon morse.

b. Metode *binary search tree* dapat diaplikasikan dengan baik pada *platform* android. Platform ini menggunakan *core* java yang dapat menerapkan struktur data *array* berurut dan senarai berantai yang merupakan hal yang mendasar dari struktur data *tree*.

c. Isyarat panjang dan pendek pada sandi morse dapat diaplikasikan menggunakan bunyi dari speaker smartphone dan sinar lampu layar smartphone dengan interval setiap titik sebesar 200 milidetik.

Adapun saran untuk melengkapi penelitian sekaligus pengembangan aplikasi ini adalah sebagai berikut:

a. Dalam penelitian ini, pohon morse yang dibangun menggunakan data sandi morse internasional yang terdiri dari huruf dan angka. Untuk itu, perlu adanya penambahan data sandi morse untuk melengkapi pohon morse dan menambahkan kunci baru untuk setiap simpul baru. Penggunaan *self balancing binary search tree dapat menggunakan AA Tree, Red-black Tree, Scapegoat Tree*, atau *Treap*.

b. Dalam penelitian ini, metode binary search tree diaplikasikan pada *platform* android. Untuk pengembangan selanjutnya, metode *binary search tree* diaplikasikan pada *platform* yang lainnya seperti *platform iphone* atau *windows mobile*.

c. Pada aplikasi ini, interval untuk penyampaian pesan morse sudah ditetapkan sehingga pengguna tidak bisa mengaturnya sesuai dengan kebutuhan. Untuk itu diperlukan adanya pengaturan yang dapat mengubah interval penyampaian pesan.

DAFTAR PUSTAKA

Dewi Rosmala, Gilang Kresna, Implementasi Algoritma Binnary Tree Pada Sistem Informasi Multilevel Marketing. Vol. 3 No.3 September-Desember 2012, Jurnal Informatika ITENAS Bandung

Guntur Syahputra, Bembi Sinurat. Impelementasi Algoritma Binnary Search Pada Kamus Indonesia Batak Toba. Vol. 1 No.1, Oktober 2016, *Journal Of Informatics Pelita Nusantara (JIPN).*

Istiyanto, Eko. 2013. *Pemrograman Smart Phone Menggunakan Sdk Android Dan Hacking Android.* Yogyakarta: Graha Ilmu.

Rinaldi Munir, Leony Lidya, 2016. Algoritma dan Pemrograman dalam Bahasa Pascal, C dan C++. Informatika Bandung.

Sjukani. 2012. *Struktur Data (Algoritma dan Struktur Data 2)*. Jakarta: Mitra Wacana Media

Sommerville. 2011. *Software Engineering (Rekayasa Perangkat Lunak)*. Jakarta: Erlangga

S, Rosa A. dam M. Shalahuddin. 2013. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Bandung: Informatika

Safaat, N. H. 2012. *Android Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android*. Bandung: Informatika.

Sianipar. 2014. *Soal dan Penyelesaian Struktur Data dengan Java*. Yogyakarta: Andi

Siswanto, 2015. *Algoritma dan Struktur Data Linier dengan Java.* Graha Ilmu

Viska Mutiawati, Juli 2014, Hashtable Sebagai Alternatif Dari Algoritma Pencarian Biner Pada Aplikasi E-Acesia, Jurnal Informatika Vol. 8, no. 2, Universitas Syiah Kuala.