

Cuprins

Listă de figuri	3
Listă de tabele	4
Capitolul 1	
INTRODUCERE	5
1.1 Actualitatea și necesitatea temei	5
1.2 Problema propusă	5
1.3 Obiectivele lucrării	5
1.4 Structura lucrării	5
Capitolul 2	
BAZA DE DATE	6
2.1 Alegerea tipului de bază de date	6
2.1.1 Factori de decizie	6
2.1.2 Soluții propuse	6
2.1.3 Soluția aleasă	7
2.2 Structura bazei de date	7
Capitolul 3	
ARHITECTURA SOFTWARE	8
3.1 Deciderea asupra comportamentului sincron și asincron	8
3.1.1 Factori de decizie	8
3.1.2 Soluții propuse	8
3.1.3 Decizia	8
3.2 Structura modului din punctul de vedere al unităților software	9
3.2.1 Unitatea software PNavServiceImpl	9
3.2.2 Unitatea software PNavImpl	9
3.2.3 Unitatea software PNavRecorder	9
3.2.4 Unitatea software PNavPredictor	9
3.2.5 Unitatea software PNavConfiguration	10
3.2.6 Unitatea software PNavCriteria	10
3.2.7 Unitatea software PNavDataManager	10

3.2.8	Unitatea software PNavDataStorage	10
Capitolul 4		
	DESCRIEREA ALGORITMILOR	12
4.1	Învățarea rutelor	12
4.1.1	Reducerea waypoint-urilor	12
4.1.2	Verificarea rutelor duble	12
4.1.3	Filtrarea rutelor inutile	13
4.2	Eliberarea de spațiu	14
4.3	Furnizarea predicțiilor	14
4.3.1	Filtrarea datelor	15
4.3.2	Frecvența predicției de rute	15
4.3.3	Numărul maxim de predicții	15
4.3.4	Prioritizarea predicțiilor	15
4.3.5	Predicții bazate pe filtrarea rutei în funcție de distanța de la poziția curentă la waypoint-urile rutelor	16
4.3.6	Predicții bazate pe filtrarea rutei în funcție de distanța de la poziția curentă la punctele de start ale rutelor	16
4.3.7	Predicții bazate pe filtrarea rutei în funcție de timpul scurs până la ajungerea la destinație	16
Capitolul 5		
	MANAGEMENTUL ERORILOR	19
5.1	Tipuri de erori	19
5.2	Detectarea erorilor	19
5.2.1	Erori de secvență	19
5.2.2	Erori la accesarea bazei de date	19
5.2.3	Bază de date coruptă	19
5.3	Tratarea erorilor	19
Capitolul 6		
	CONCLUZII ȘI DIRECȚII VIITOARE	21
6.1	Concluzii generale	21
6.2	Posibilități de dezvoltare ulterioară	21

Listă de figuri

2.1	Structura tabelelor de date și relațiile dintre ele	7
4.2	Procesul de verificarea al rutelor duble (varianta sincronă)	13
4.3	Procesul de ștergere a uneia sau mai multor rute (varianta sincronă)	14

Listă de tabele

2.1	Compararea principalelor metode de stocare a datelor pe baza factorilor de influențare	6
4.2	Criterii de prioritizare pentru predicția bazată pe rute	17
4.3	Criterii de prioritizare pentru predicția bazată pe rutele din jurul unei poziții	17
4.4	Criterii de prioritizare pentru predicția bazată pe filtrarea rutele în funcție de distanța până la punctul de start al rutei	18

Capitolul 1

INTRODUCERE

1.1 Actualitatea și necesitatea temei

1.2 Problema propusă

1.3 Obiectivele lucrării

1.4 Structura lucrării

1. Introducere
2. Baza de date
3. Arhitectura software
4. Descrierea algoritmilor
5. Gestionarea erorilor
6. Concluzii și dezvoltări ulterioare

Capitolul 2

BAZA DE DATE

2.1 Alegerea tipului de bază de date

Nevoia modulului de a stoca și de a accesa datele stocate anterior a dus la realizarea unui mic studiu în vederea alegerii tipului de bază de date cel mai potrivit.

2.1.1 Factori de decizie

- Consistența datelor stocate
- Utilizarea RAM-ului
- Timp de accesare la pornirea aplicației
- Timp de accesare în cadrul aplicației
- Spațiul ocupat pe disc
- Compatibilitate cu versiunile anterioare

2.1.2 Soluții propuse

În următorul tabel, se presupune ca pentru fișierele binare, XML și JSON este necesară încărcarea datelor la pornirea aplicației. SQLite oferă însă soluții de căutare inteligente, nefiind necesară încărcarea tuturor datelor la pornirea aplicației.

TABELA 2.1: Compararea principalelor metode de stocare a datelor pe baza factorilor de influențare

	Binar	XML sau JSON	SQLite	Memorare în Cloud
Consistența datelor stocate	Nu	Nu	Da	Da
Utilizarea RAM-ului	Ridicat	Scăzut	Mediu	Mediu
Timp de accesare la pornirea aplicației	Mediu	Ridicat	Scăzut	Scăzut
Timp de accesare în cadrul aplicației	Scăzut	Scăzut	Mediu	Ridicat
Spațiul ocupat pe disc	Scăzut	Ridicat	Scăzut	Foarte scăzut
Compatibilitate cu versiunile anterioare	Nu	Da	Da	Nu

2.1.3 Soluția aleasă

S-a decis folosirea SQLite ca format pentru baza de date deoarece îndeplinea toate criteriile specificate.

2.2 Structura bazei de date

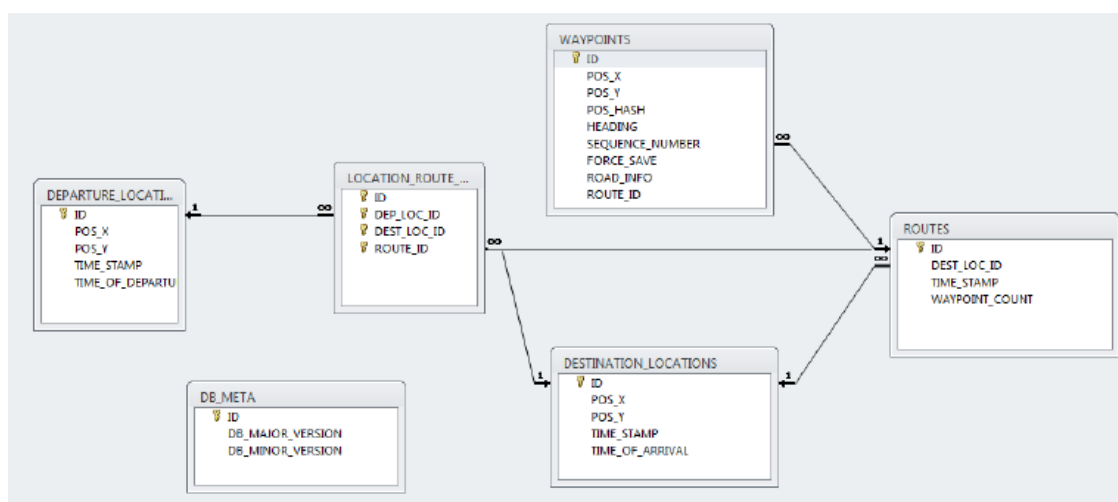


FIGURA 2.1: Structura tabelor de date și relațiile dintre ele

Tabelele din figura de mai sus sunt folosite pentru realizarea structurii întregii baze de date.

Tabela meta este folosită la identificarea versiunii bazei de date. Acest lucru este necesar pentru a detecta compatibilitatea și pentru a permite migrarea către o versiune mai recentă.

Datele înregistrate sunt separate în puncte de plecare, destinații, rute și waypoint-uri. O rută este întotdeauna formată din mai multe waypoint-uri, unul sau mai multe puncte de plecare și una sau mai multe destinații. Ruta (waypoint-urile) sunt stocate numai o singură dată, în timp ce toate punctele de plecare și destinațiile sunt stocate. În acest fel, numărul de destinații poate influența probabilitatea rutei.

Accesul la date se face prin SQLite. Toate datele stocate pot fi atât citite cât și modificate.

Capitolul 3

ARHITECTURA SOFTWARE

În acest capitol este prezentată partea structurală a modulului cât și unitățile software din care acesta este format.

3.1 Deciderea asupra comportamentului sincron și asincron

Fiecare mod de executare a operațiilor are propriile sale avantaje și dezavantaje. Există operații care ar putea necesita mai mult timp pentru a se termina de executat ($>100\text{ms}$) și nu este direct vizibil faptul dacă acestea au fost declanșate de către o cerere (e.g. o nouă poziție este trimisă).

O cerere poate declanșa din fire de execuție diferite. Există posibilitatea ca acest lucru să fie realizat sincron, în afara modului, între cereri și răspunsuri, fapt ce poate duce la deadlock.

3.1.1 Factori de decizie

- Timpul în care firul de execuție este blocat de cerere
- Sincronizarea între operații

3.1.2 Soluții propuse

- Procesul se va executa asincron folosind un fir de execuție de lucru
- Procesul se va executa sincron, în interiorul cererilor

3.1.3 Decizia

Se vor furniza două interfețe diferite. Funcționalitatea va fi oferită printr-o interfață sincronă, ce va fi utilizată în cadrul operațiilor ce au loc pe un singur fir de execuție. O altă interfață va decupla firele de execuție și procesele din bucla de lucru. Acest fapt ne oferă libertatea utilizării principiului de multithread-ing (execuția mai multor thread-uri în același pipeline, fiecare având propria secțiune de timp în care este menit să lucreze).

3.2 Structura modului din punctul de vedere al unităților software

3.2.1 Unitatea software PNavServiceImpl

Unitatea PNavServiceImpl implementează interfața sincronă și asincronă, și îi oferă totodată dezvoltatorului posibilitatea de a alege ce interfață dorește să folosească.

Cea asincronă are avantajul de a decupla firele de execuție și de a permite rularea activităților în paralel, însă are și dezavantajul necesității de implementare unui mecanism de sincronizare în codul aplicației în care va fi folosit modulul.

3.2.2 Unitatea software PNavImpl

Deși funcționalitățile de bază precum învățarea și predicția sunt realizate de către unitatea PNavRecorder respectiv PNavPredictor, unitatea PNavImpl realizează funcționalități suplimentare cum ar fi multiple profile de utilizatori, ștergerea bazelor de date.

Funcționalitatea multiplelor profile de utilizatori permite gestionarea mai multor baze de date, ce pot fi selectate pe baza unui ID de profil. Acest ID poate cuprinde valori în intervalul 0-255. Pentru fiecare profil este creat un nou fișier în care vor fi stocate datele de utilizator. Unitatea PNavImpl implementează de asemenea și funcționalități de întreținere a profilelor de utilizator precum ștergerea individuală, ștergerea totală, copierea, schimbarea între profile.

3.2.3 Unitatea software PNavRecorder

NavRecorder-ul este unitate în care întreg procesul de învățare are loc.

Unitatea primește datele de geolocație și de timp (oră - zi/lună/an) și învață rutele parcurse de către dezvoltator într-un mod inteligent. Acest lucru înseamnă că waypoint-urile (punctele prin care a trecut utilizatorul în timpul rutei sale) sunt stocate numai când autovehiculul și-a schimbat orientarea semnificativ (valoare standard: $> 15^\circ$) sau distanța dintre waypoint-uri nu este prea scurtă (valoare standard: $> 200\text{m}$). Valori pot fi configurate înaintea procesului de compilare.

Când sesiunea de înregistrare este finalizată, waypoint-urile sunt trimise către unitatea software PNavDataManager pentru a fi scrise în baza de date. Totodată, unitatea are implementate funcționalități de oprire-pornire, lucru ce-i acordă dezvoltatorului dreptul de opri și porni oricând sesiunea de înregistrare.

3.2.4 Unitatea software PNavPredictor

Rolul unității PNavPredictor este acela de a calcula predicțiile.

Primul pas constă în încărcarea datelor prin unitatea PNavDataManager, care sunt mai departe prioritizate în funcție de criterii specifice (descrise în tabela 4.2, “Criterii de prioritizare pentru predicția bazată pe rute”). Datele pot fi de asemenea filtrate pe baza aceluiași criterii, rezultând astfel o cantitate mai mică de date și un timp mai scurt de încărcare a acestora.

Prioritizarea datelor este bazată atât pe datele de geolocație cât și cele de timp, astfel încât o rută va avea o probabilitate mult mai mare de utilizare într-o anumită zi din săptămână sau la o anumită oră din zi.

Ca și unitatea PNavRecorder, unitatea PNavPredictor are implementate funcționalități de oprire-pornire.

3.2.5 Unitatea software PNavConfiguration

Unitatea PNavConfiguration configurează unitatea NavPredictor, prin intermediul unor funcții ce folosesc criteriile definite în unitatea PNavCriteria.

3.2.6 Unitatea software PNavCriteria

Unitatea PNavCriteria conține toate tipurile de criterii ce pot fi folosite la filtrarea sau prioritizarea datelor.

Fiecare criteriu în parte este folosit la procesarea datelor de către unitatea PNavPredictor. După procesarea tuturor criteriilor cea mai probabilă rută este creată.

3.2.7 Unitatea software PNavDataManager

Unitatea PNavDataManager implementează logica necesară pentru a realiza comunicarea între unitatea PNavDataStorage și restul unităților.

În general, obiectele sunt stocate separat (e.g. rutele sunt stocate separat față de destinațiile lor). Cum însă pentru predicția unei rute este nevoie de toate informațiile, unitatea PNavDataManager le comasează. Aceasta oferă de asemenea și alte funcționalități precum adăugarea, gruparea, căutarea sau ștergerea de obiecte.

3.2.8 Unitatea software PNavDataStorage

Unitatea PNavDataStorage este dezvoltată pe baza structurii bazei de date.

În afară de funcționalitatea principală de a stoca sau încărca date, aceasta asigură și accesarea selectivă a obiectelor. Pentru realizarea acestor funcționalități se execută interogări prin intermediul SQLite.

Capitolul 4

DESCRIEREA ALGORITMILOR

4.1 Învățarea rutelor

Rutele sunt învățate printr-un mecanism inteligent. Acest lucru are avantajul de a reduce semnificativ baza de date și de a accelera încărcarea datelor.

4.1.1 Reducerea waypoint-urilor

Criteriile pentru excluderea waypoint-urilor sunt:

1. Vehiculul nu și-a schimbat orientarea semnificativ (valoare standard: $> 15^\circ$)
2. Pozițiile sunt apropiate una de cealaltă (valoare standard: $> 200\text{m}$)

Valorile standard pot fi configurate înaintea procesului de compilare.

4.1.2 Verificarea rutelor duble

În cazul în care ruta curentă are o secțiune similară cu o rută deja învățată, unitatea software PNavDataManager va detecta secțiune respectivă și va stoca numai waypoint-urile situate după aceasta. În schimb, toate destinațiile sunt memorate. Criteriile pentru a detecta o astfel de rută sunt:

1. Destinația noii rute trebuie să fie situată într-o rază de 1km față de locația vechii destinații
2. Punctul de start al noii rute trebuie să fie situată într-o rază de 1km față de punctul de plecare vechii destinații
3. Toate waypoint-urile noii rute trebuie:
 - (a) Să fie situat într-o anumită rază față de punctul de start al vechii rute
 - (b) Sau să fie situat într-o anumită rază față de destinația vechii rute
 - (c) Sau să fie situat într-o anumită rază față de un waypoint al vechii rute

Pragurile pot fi configurate înaintea procesului de compilare.

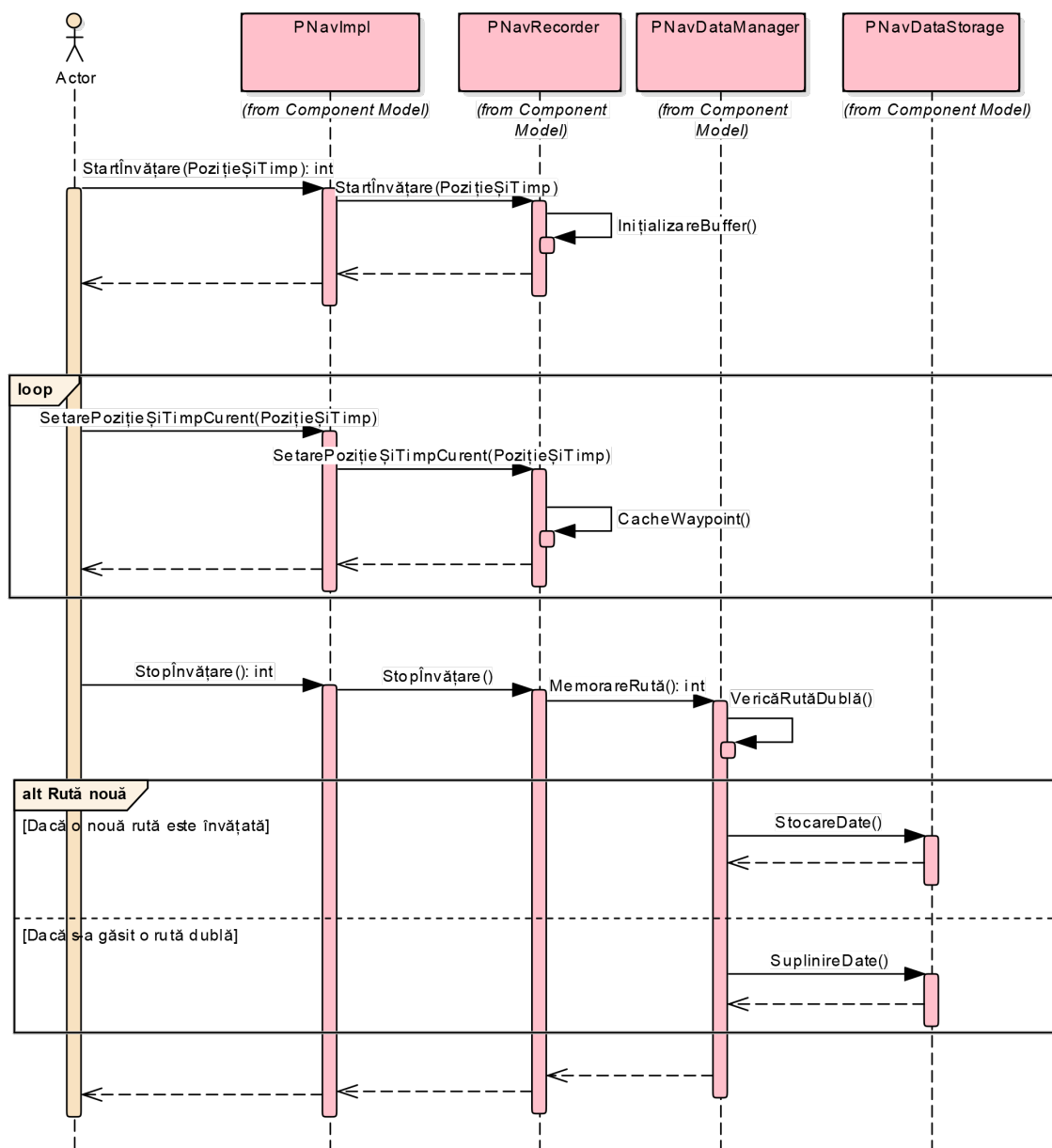


FIGURA 4.2: Procesul de verificarea al rutelor duble (varianta sincronă)

4.1.3 Filtrarea rutelor inutile

În timpul învățării, rutele prea scurte ($< 2\text{km}$) nu vor fi stocate deoarece acest lucru ar însemna faptul că utilizatorul este destul de aproape de destinație.

Totodată, rutele foarte lungi ($> 200\text{km}$) vor fi de asemenea excluse din stocare deoarece ele nu reprezintă rute uzuale.

Pragurile pot fi configurate înaintea procesului de compilare.

4.2 Eliberarea de spațiu

Dacă pragul setat inițial pentru dimensiunea maximă a bazei de date este atins, se va activa funcția de eliberare a spațiului. Aceasta va șterge obiectele cele mai vechi pentru a crea loc pentru obiectele noi.

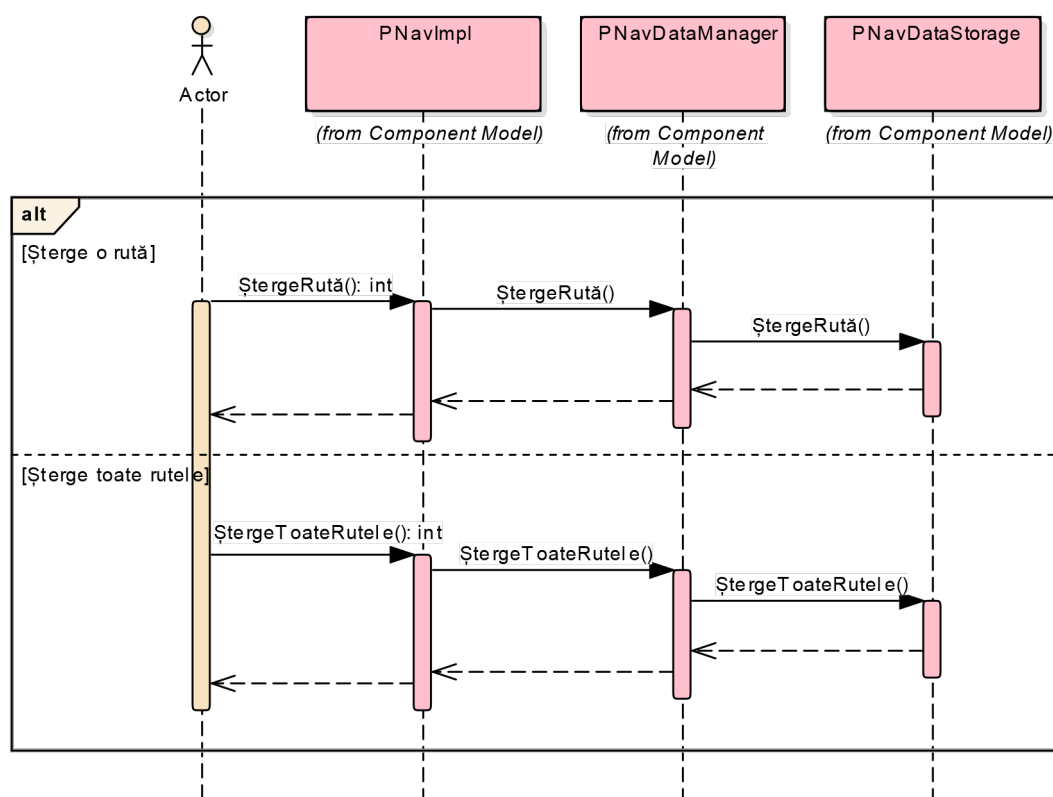


FIGURA 4.3: Procesul de ștergere a uneia sau mai multor rute (varianta sincronă)

4.3 Furnizarea predicțiilor

Pentru ca predicțiile să fie disponibile sunt necesari doi pași.

Primul pas reprezintă încărcarea datelor relevante, în timp ce al doilea constă în prioritizarea lor.

4.3.1 Filtrarea datelor

Pentru o încărcare selectivă și mai rapidă a datelor din unitatea PNavDataStorage, sunt create interogări. Sunt definiți trei pași în filtrare, unde pasul următor se execută doar în cazul în care cel curent nu a returnat destule rezultate:

1. Filtrarea rutei în funcție de distanța de la poziția curentă la waypoint-urile rutelor
2. Filtrarea rutei în funcție de distanța de la poziția curentă la punctele de start ale rutelor
3. Filtrarea rutelor în funcție de timpul scurs până la ajungerea la destinație

Toate rutele ce duc la o destinație situată la o distanță mai mică de 1km față de poziția actuală sunt ignorate deoarece utilizatorul aproape a ajuns la eventuala destinație.

În cazul predicțiilor baza pe timp, modulul furnizează o predicție fără a cunoaște ruta, ci doar destinația sa. Un astfel de caz ar fi cel în care utilizatorul a condus pe o rută în intervalul luni-miercuri, însă în ziua de joi a pornit de la o altă locație. Astfel, bazat pe timp, modulul prezice destinația fără a ști ruta corespunzătoare acesteia.

4.3.2 Frecvența predicției de rute

De obicei, waypoint-urile furnizate de modul de predicție sunt transformate într-o rută ce folosește drumuri din hartă, fapt ce durează câteva secunde. Pentru a nu supraîncărca sistemul de navigație cu prea multe predicții, dezvoltatorul poate seta timpul minim dintre două predicții. Această setare se poate efectua chiar în timpul rulării.

4.3.3 Numărul maxim de predicții

Pentru a putea suporta multiple platforme, modulul permite setarea numărului maxim de rute prezise ce apar deodată. Această setare se poate efectua chiar în timpul rulării.

4.3.4 Prioritizarea predicțiilor

Prioritatea unei rute reprezintă de fapt de probabilitatea ca această rută să fie ce pe care utilizatorul ar fi ales-o în mod uzual. Aceasta este exprimată sub formă de procente și se încadrează în intervalul 0%-100%.

Există mai multe criterii pentru a stabili prioritatea unei rute, fiecare dintre acestea fiind raportat la intervalul 0 (minimul) - 100 (maximul).

Pentru calcularea probabilității unei rute sunt necesari trei pași:

1. Pentru fiecare criteriu, se înmulțește ponderea cu probabilitatea sa

$$SC = \sum_{i=1}^n p_i * c_i \quad (4.1)$$

2. Se însumează toate criteriile

$$SP = \sum_{i=1}^n p_i \quad (4.2)$$

3. În urma raportului dintre suma de la pasul 1 și suma de la pasul 2, se obține probabilitatea rutei

$$P = \frac{SC}{SP} \quad (4.3)$$

unde, P = probabilitatea rutei, SC = suma criteriilor, SP = suma ponderilor, p = ponderea criteriului, c = probabilitatea criteriului (0%-100%), n = numărul total de criterii

Metoda de definire a fiecărui criteriu depinde de criteriile însăși, după cum se observă în tabelele următoare.

4.3.5 Predicții bazate pe filtrarea rutei în funcție de distanța de la poziția curentă la waypoint-urile rutelor

Criteriile definite în tabela 4.2, “Criterii de prioritizare pentru predicția bazată pe rute” sunt extinse prin adăugarea următoarelor criterii:

4.3.6 Predicții bazate pe filtrarea rutei în funcție de distanța de la poziția curentă la punctele de start ale rutelor

Criteriile definite în tabela 4.2, “Criterii de prioritizare pentru predicția bazată pe rute” sunt extinse prin adăugarea următoarelor criterii:

4.3.7 Predicții bazate pe filtrarea rutei în funcție de timpul scurs până la ajungerea la destinație

Pentru acest caz sunt folosite criteriile definite în tabela 4.2, “Criterii de prioritizare pentru predicția bazată pe rute”.

TABELA 4.2: Criterii de prioritizare pentru predicția bazată pe rute

Criteriu	Descriere	Min (0% probabilitate)	Max (100% probabilitate)	Pondere
Frecvența rutei	Numărul de utilizări al unei rute	Niciodată	Folosită de mai mult de 10 ori	4
Frecvența rutei într-un anumit interval de timp	Numărul de utilizări al unei rute într-un anumit interval de timp (de la -1 oră la +2 ore) față de ora curentă	Niciodată	Folosită de mai mult de 10 ori	1
Frecvența rutei într-o anumită zi	Numărul de utilizări al unei rute în ziua curentă din săptămână	Niciodată	Folosită de mai mult de 10 ori	1
Frecvența rutei într-un anumit grup de zile	Numărul de utilizări al unei rute într-un anumit grup de zile (e.g. luni-vineri)	Niciodată	Folosită de mai mult de 10 ori	1
Ultima utilizare a unei rute	Diferența dintre ultima utilizare a rutei și ora/data curentă	>4 săptămâni	<2zile	3
Distanța până la destinație	Distanța dintre poziția actuală și destinație	>100km	0km	1

TABELA 4.3: Criterii de prioritizare pentru predicția bazată pe rutele din jurul unei poziții

Criteriu	Descriere	Min (0% probabilitate)	Max (100% probabilitate)	Pondere
Distanța până la rută	Distanța dintre poziția actuală și waypoint-urile rutei	> 1km	0km	1
Direcția către rută	Diferența dintre orientarea waypoint-urilor și direcția de navigare	180°	0°	1

TABELA 4.4: Criterii de prioritzare pentru predicția bazată pe filtrarea rutele în funcție de distanța până la punctul de start al rutei

Criteriu	Descriere	Min (0% probabilitate)	Max (100% probabilitate)	Pondere
Distanța până la punctul de start al rutei	Distanța până la cel mai apropiat punct de start al rutei	> 3km	0km	1

Capitolul 5

MANAGEMENTUL ERORILOR

5.1 Tipuri de erori

- Erori de secvență
- Erori apărute la accesarea bazei de date
- Bază de date coruptă

5.2 Detectarea erorilor

Erorile sunt raportate pentru fiecare apel către și dinspre interfața modului de predicție. Toate funcțiile returnează un număr ce corespunde unui tip de eroare.

5.2.1 Erori de secvență

O mașină de stare va verifica încălcarea ordinii secvențelor.

5.2.2 Erori la accesarea bazei de date

Trebuie evaluate rezultatele funcțiilor native de accesare ale bazei de date.

5.2.3 Bază de date coruptă

Trebuie evaluate rezultatele funcțiilor native de accesare ale bazei de date.

5.3 Tratarea erorilor

În funcție de tipul de eroare, aceasta poate fi prevenită pe viitor sau nu de către dezvoltator.

Există erori de secvență precum “Înainte de apelarea funcției X este necesară pornirea unității software PNavPredictor”. Se poate întâmpla de asemenea ca atunci când dezvoltatorul pornește procesul de predicție de două ori consecutiv să fie întâmpinat de eroarea “Procesul de predicție se

află deja în curs de rulare”. În astfel de cazuri este clar cum se pot preveni erorile.

Mai sunt însă și cazuri ce nu pot fi tratate de către dezvoltator. Astfel de erori sunt cele precum “Baza de date este coruptă”, ce pot să apară în cazul în care fișierul de sistem folosit pentru stocarea datelor este corupt. În aceste situații datele stocate anterior nu mai pot fi recuperate, toate informațiile referitoare la rute fiind definitiv pierdute.

Capitolul 6

CONCLUZII ȘI DIRECȚII VIITOARE

6.1 Concluzii generale

6.2 Posibilități de dezvoltare ulterioară