**Sistem distribuit de mesaje**

Întocmit de: **Balog David Alexandru**

**Ossian Mihai Vlad**

**Marchis Ionuț Florin**

Pentru disciplina: **Sisteme Distribuite**

Data predării: **13//01//2025**

Cuprins

[Capitolul 1. Introducere 1](#_Toc384994105)

[1.1. Contextul proiectului 1](#_Toc384994106)

[1.2. Scopul proiectului 1](#_Toc384994106)

[Capitolul 2. Componente și funcționalități ale sistemului 2](#_Toc384994105)

[2.1. Componente principale 2](#_Toc384994106)

[2.2. Funcționalități implementate 2](#_Toc384994106)-3

[2.3. Tehnologii utilizate 3](#_Toc384994106)

[Capitolul 3. Avantaje și limitări 4](#_Toc384994105)

[Capitolul 4. Obiective, etape și dificultăți 5](#_Toc384994105)

[4.1. Obiective 5](#_Toc384994106)

[4.2. Etapele planificate 5](#_Toc384994106)

[4.3. Dificultăți și propuneri 5](#_Toc384994106)

[4.3.1. Dificultăți 1](#_Toc384994107)

[4.3.2. Propuneri 1](#_Toc384994107)

[Bibliografie 5](#_Toc384994115)

[Anexa 1 (dacă este necesar) 5](#_Toc384994116)

# Introducere

## Contextul proiectului

Sistemele distribuite reprezintă o componentă esențială a infrastructurii tehnologice moderne, oferind soluții scalabile și reziliente pentru gestionarea datelor și resurselor. Acest document descrie implementarea unui sistem distribuit utilizând protocolul TCP (Transmission Control Protocol), un protocol de transport fiabil și orientat pe conexiune, care asigură comunicarea între noduri într-un mod robust și eficient.

Scopul acestui sistem este de a distribui fișiere și date între noduri printr-o rețea logică inelară, utilizând TCP pentru a asigura transferuri corecte, ordonate și fără pierderi de date. Implementarea include funcționalități precum detectarea și gestionarea eșecurilor nodurilor, replicarea datelor pentru redundanță, și rutarea eficientă a cererilor.

Protocolul TCP permite acestui sistem să asigure fiabilitatea comunicării între noduri, fiind ideal pentru aplicații care necesită consistență ridicată a datelor și garanții privind livrarea pachetelor. Acest sistem este potrivit pentru utilizarea în rețele descentralizate, precum platforme de partajare a fișierelor, rețele blockchain sau alte aplicații distribuite similare.

## Scopul proiectului

Proiectul are ca obiectiv principal dezvoltarea unui sistem distribuit capabil să asigure:

* Distribuirea datelor între noduri într-un mod echitabil
* Rezistența la căderea unor noduri din inel, prin utilizarea replicării datelor și a unor mecanisme de rerutare eficiente.
* Gestionarea automată a eșecurilor unui nod picat, prin detectarea rapidă a problemelor și ajustarea structurii inelului fără a compromite funcționarea sistemului.

# Componente și funcționalități ale sistemului

## Componente principale

1. Nodurile:
   * Identificator unic generat prin id.
   * Responsabilități: stocarea și redistribuirea datelor catre alte noduri si clienti.
2. Protocolul de comunicare:
   * Se bazează pe TCP (Transmision Control Protocol) pentru schimbul de mesaje între noduri.
3. Detectarea eșecurilor:
   * Mecanism de ping periodic („heartbeat”) pentru monitorizarea nodurilor vecine.
   * Mecanism de redirecționare a nodului urmator in mesajului heartbeat.
4. Server nod
   * Un nod similar cu cele din inel care trimite date noastre unui nod aleatoriu.

## Funcționalități implementate

1. Adăugare de noduri. Atunci când un nod nou se alătură inelului:

* Nodului i se precizeaza adresa/portul său, adresa/portul urmatorului nod în inel și id-ul acestuia.
* În cazul în care acesta nu se poate conecta la următorul nod, el v-a incerca să se conecteze la noduri incrementate.

1. Transferul de mesaje.

* Mesajele sunt distribuite in inel de catre nodul server.

1. Detectarea și gestionarea eșecurilor

* Eșecurile sunt detectate prin lipsa răspunsurilor la ping-urile heartbeat regulate trimise succesorilor și predecesorilor.
* Structura inelului este ajustată automat pentru a exclude nodurile eșuate.

## Tehnologii utilizate

* Limbajul de programare: Java
* Mediu de dezvoltare: Visual Studio Code
* Biblioteci și framework-uri: Jackson .

# Avantaje și limitări

1. Avantaje
   * Scalabilitate: Sistemul poate gestiona eficient adăugarea sau eliminarea unui număr mare de noduri.
   * Robustețe: Replicarea asigură continuitatea serviciilor în cazul eșecurilor nodurilor.
2. Limitări
   * Latența rețelei: Performanța poate varia în funcție de latența între noduri.
   * Consum de resurse: Replicarea extinsă poate consuma resurse suplimentar
   * Complexitate: Gestionarea eșecurilor multiple simultane poate deveni dificilă.

# Obiective, etape și dificultăți

## Obiective

Obiectivele propuse de noi sunt:

* + Categorii,preferințe

Știrile vor avea un parametru string sau list string pentru care vom efectua o filtrare.

* + Algoritm pentru filtrare a preferințelor
  + Clasă știri cu multiple categorii

Unele știri vor avea multiple categorii/hashtaguri pentru acces la o audientă mai mare, mai amplă.

* + Server cu threding

Serverul care va primi JSON-urile de la API și le va ruta către cerere (request).

* + Rest API

Partea de fetching a JSON-ului din API (prelucrare dacă este nevoie). API-ul pentru știri: https://mediastack.com/

* + Salvat de știri în fișierul JSON/text la receiver

Atunci când se va introduce o știre nouă cererile care au tag-ul ales de utilizator, vor primi și vor salva pe sistem.

## Etapele planificate

Etapa 1 (25.nov-10.dec) – Lucrăm la partea funcționala de trimitere a mesajelor și primire

Etapa 2 (10.dec-24.dec) – Trimitem mesajele cu conținutul din API + salvarea mesajelor primite (receiver)

Etapa 3 (24.dec-31.dec) – Adăugare filtru pentru cereri și partea de testare a sistemului

Etapa 4 (31.dec-13.ianuarie) – Crearea documentației și organizarea codului sursă

## Dificultăți și propuneri

### Dificultăți

Fiecare participant ar trebui să primească toate știrile existente. În acest caz vom avea aceleași informații la toți participanți la comunicare.

În cazul în care fiecare participant la conversație va primi doar știrile filtrate de ei, în cazul apariției unui nou participant, acesta va putea trimite doar știrile filtrate de către participantul 1 (pe noul participant nu îl interesează această știre).

### Propuneri

Propunere 1: Fiecare participant va salva doar știrea sub interesul acesteia iar în apariția unui nou participant acesta o să îi ofere doar aceste știri.

Propunere 2: Toți participanți vor primi toate știrile și se vor salva toate știrile, la apariția unui nou participant acesta va primi toate știrile și le va comunica tuturor participanților pentru actualizarea acestora.

Propunere 3: Actualizarea de știri v-a fi periodică.

## Contribuția echipei

Fiecare membru al echipei a contribuit activ la realizarea proiectului, participând în mod constant la întâlnirile organizate. Prin colaborare și comunicare eficientă, echipa a reușit să identifice soluții optime pentru fiecare provocare întâlnită.

# Bibliografie

<http://cti.ubm.ro/sdi/>

<https://kb.cunbm.utcluj.ro/course/view.php?id=2637>

# Anexe

**1. Cod sursă**

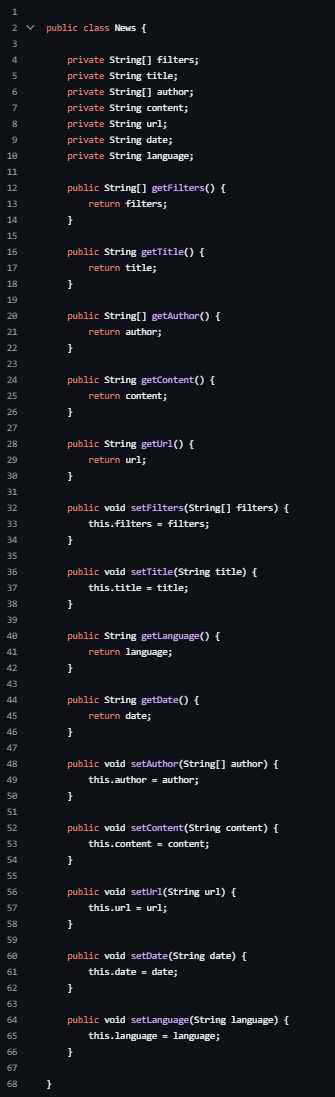
1. Client.java



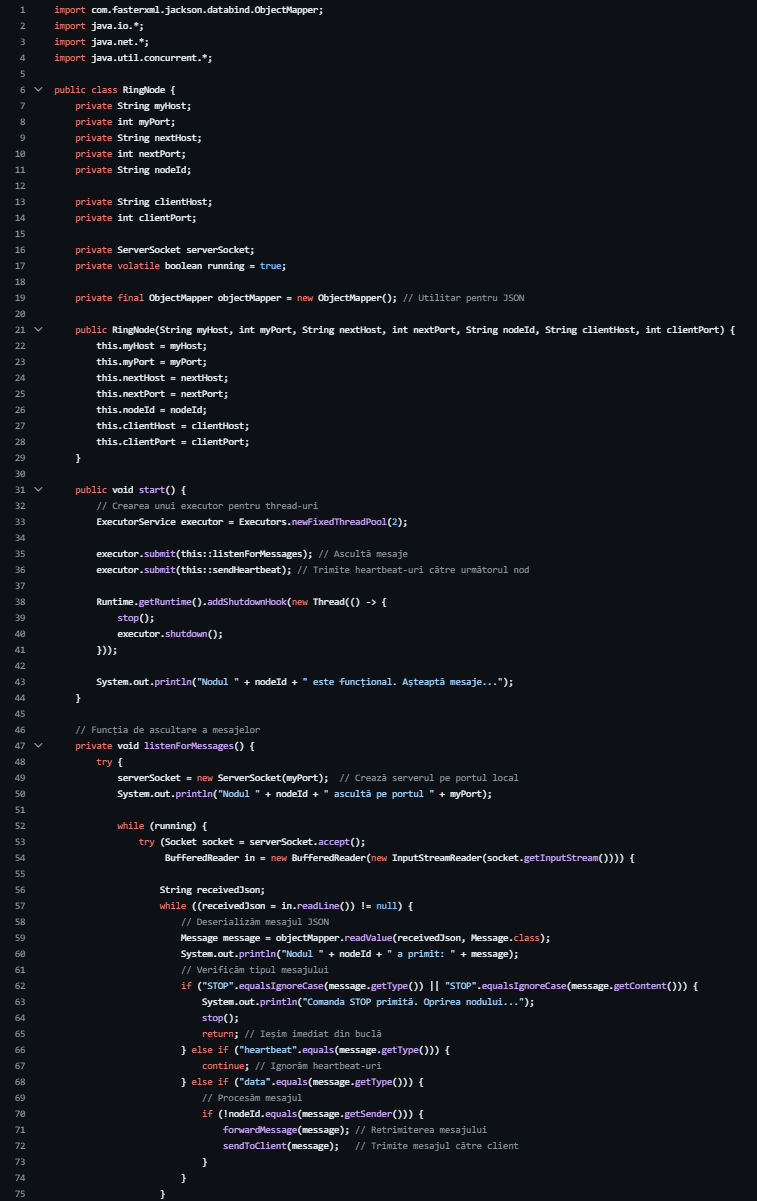
1. Message.java

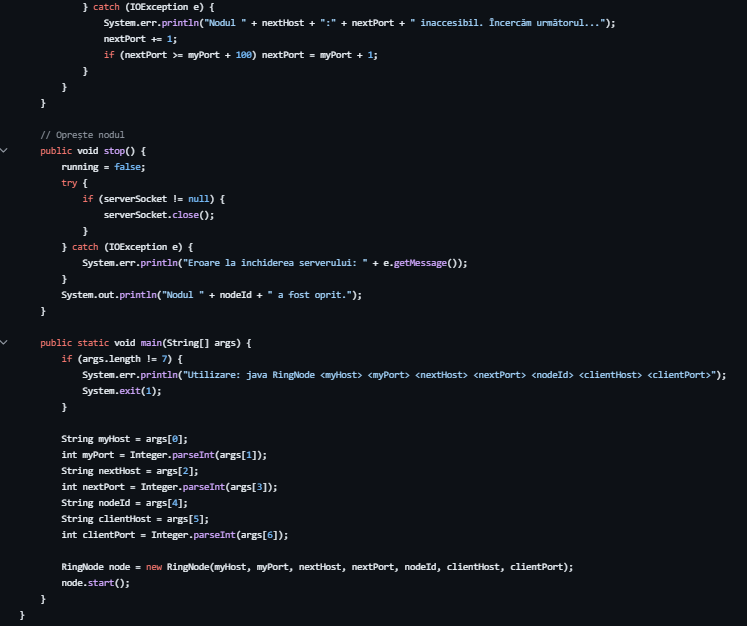


1. News.java



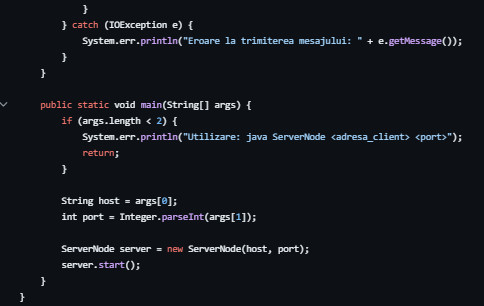
1. RingNode.java



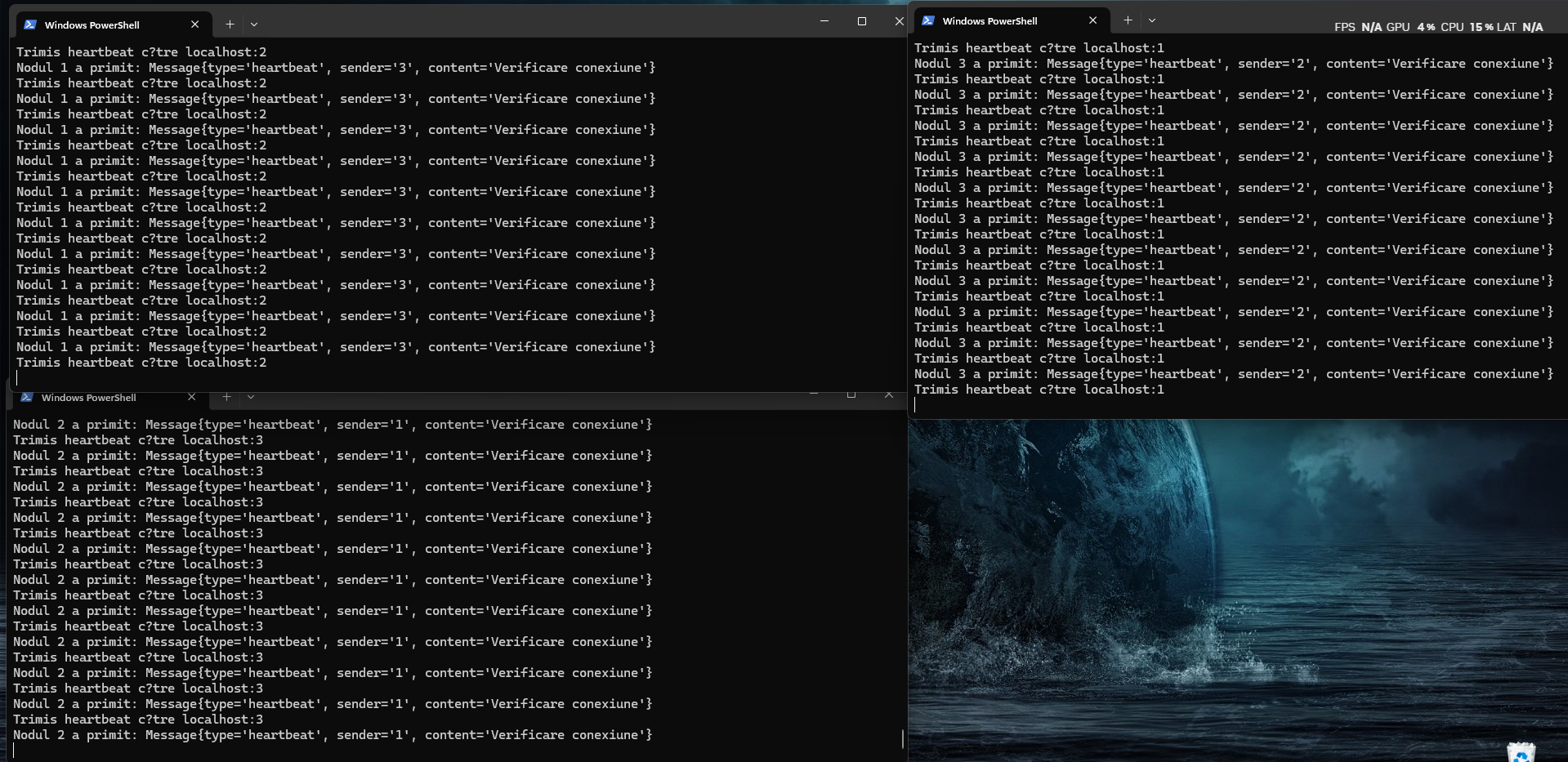


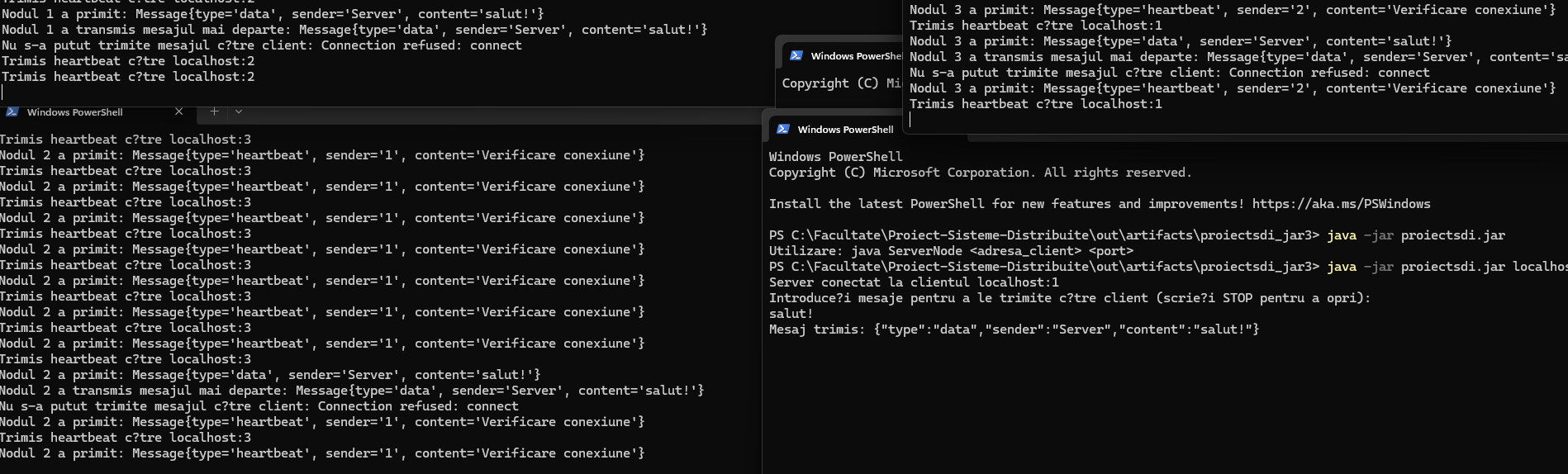
1. ServerNode.java





1. **Rulare aplicație**

****

****