# 

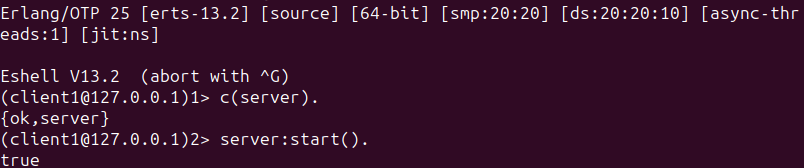
# **SISTEMES OPERATIUS DISTRIBUÏTS I EN XARXA**

**Chatty: a simple chat service**

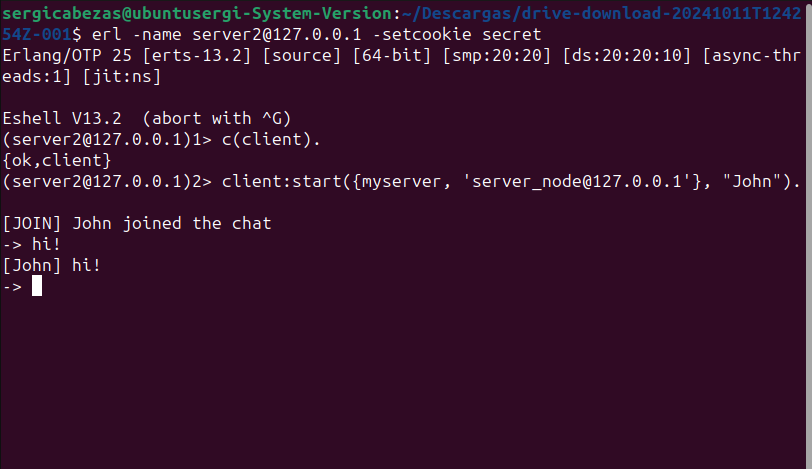
**PART 1**

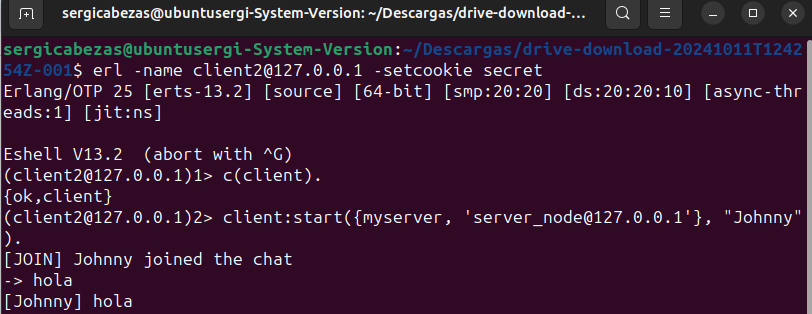
**Experiments**

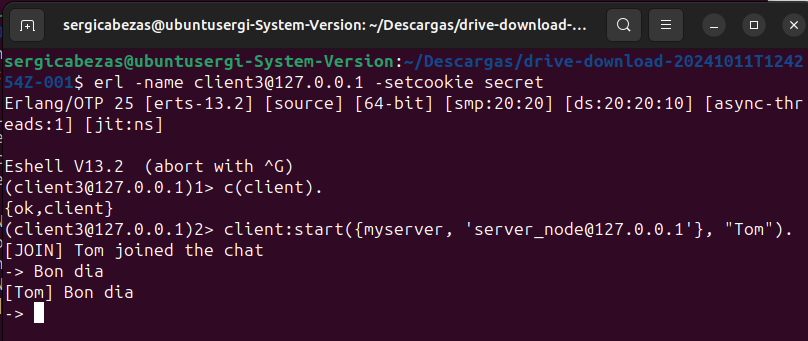
1. Compilem el servidor i l’iniciem



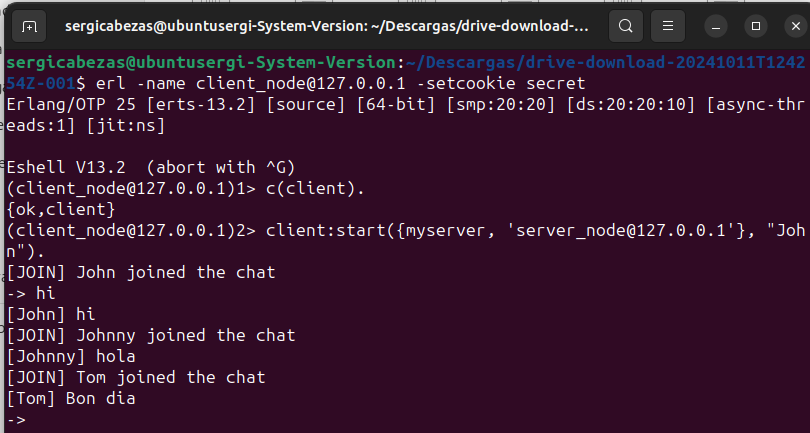
1. En una altre terminal entrem el primer usuari del xat

****

1. Al punt 3 i 4 repetim el punt 2 inician un altre usuari del xat diferent



1. Comprovem que el xat funcioni correctament

****

**Open Questions.**

**i) Does this solution scale when the number of users increase?**

No, aquesta solució no és escalable quan el nombre d'usuaris augmenta. Els missatges han de passar per un únic punt, el que provoca que el servidor es torni més lent a mesura que augmenta el nombre de clients. Cada vegada que

s'envia un missatge pot generar l’acumulació de missatges en cua i acabar amb bloquejos, especialment si el nombre de clients augmenta considerablement.

**ii) What happens if the server fails?**

En cas que el servidor falli tots els clients connectats perdran la connexió, sense resposta, i amb això tota possibilitat de comunicar-se, ja que el servidor és responsable de mantenir els clients connectats i retransmetre missatges i és l'únic que tenim.

**iii) Are the messages from a single client guaranteed to be delivered to any other client in the order they were issued? (hint: search for the 'order of message reception' in Erlang FAQ)**

Si Erlang garanteix l'ordre de missatges entre 2 processos específics, per tant, sí que arribaran en el mateix ordre que van ser enviats.

**iv) Are the messages sent concurrently by several clients guaranteed to be delivered to any other client in the order they were issued?**

No es garanteix que arribin en el mateix ordre que van ser enviats, el servidor retransmetrà els missatges en l'ordre en què els rep, però hi poden haver diferències, el qual provoca que els missatges puguin arribar en un diferent ordre.

**v) Is it possible that a client receives a response to a message from another client before receiving the original message from a third client?**

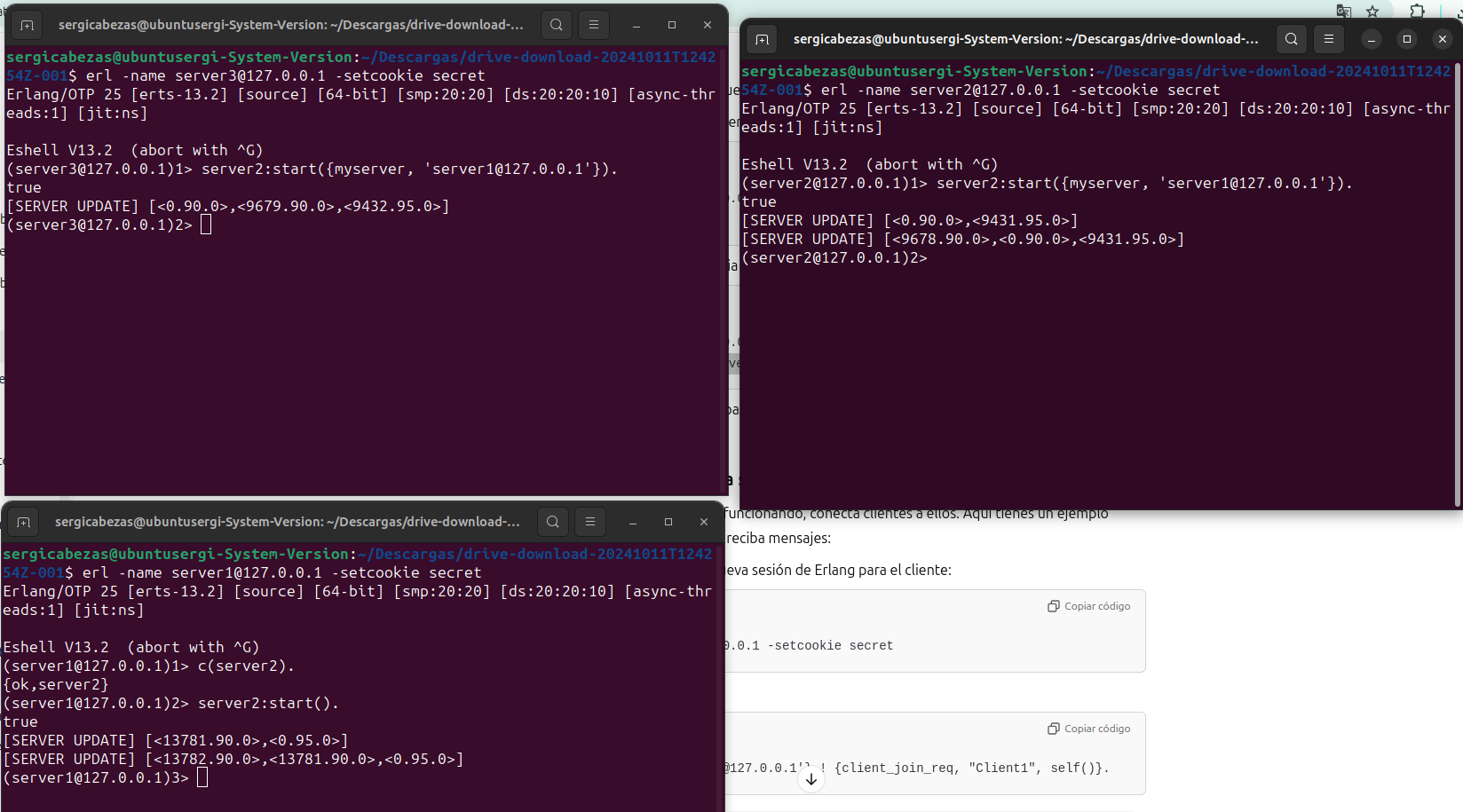
Sí, és possible. Donat que els missatges enviats per diferents clientes al servidor poden ser processats i retransmets intercaladament, un client podria rebre una resposta abans de rebre el missatge original d'un altre client**.**

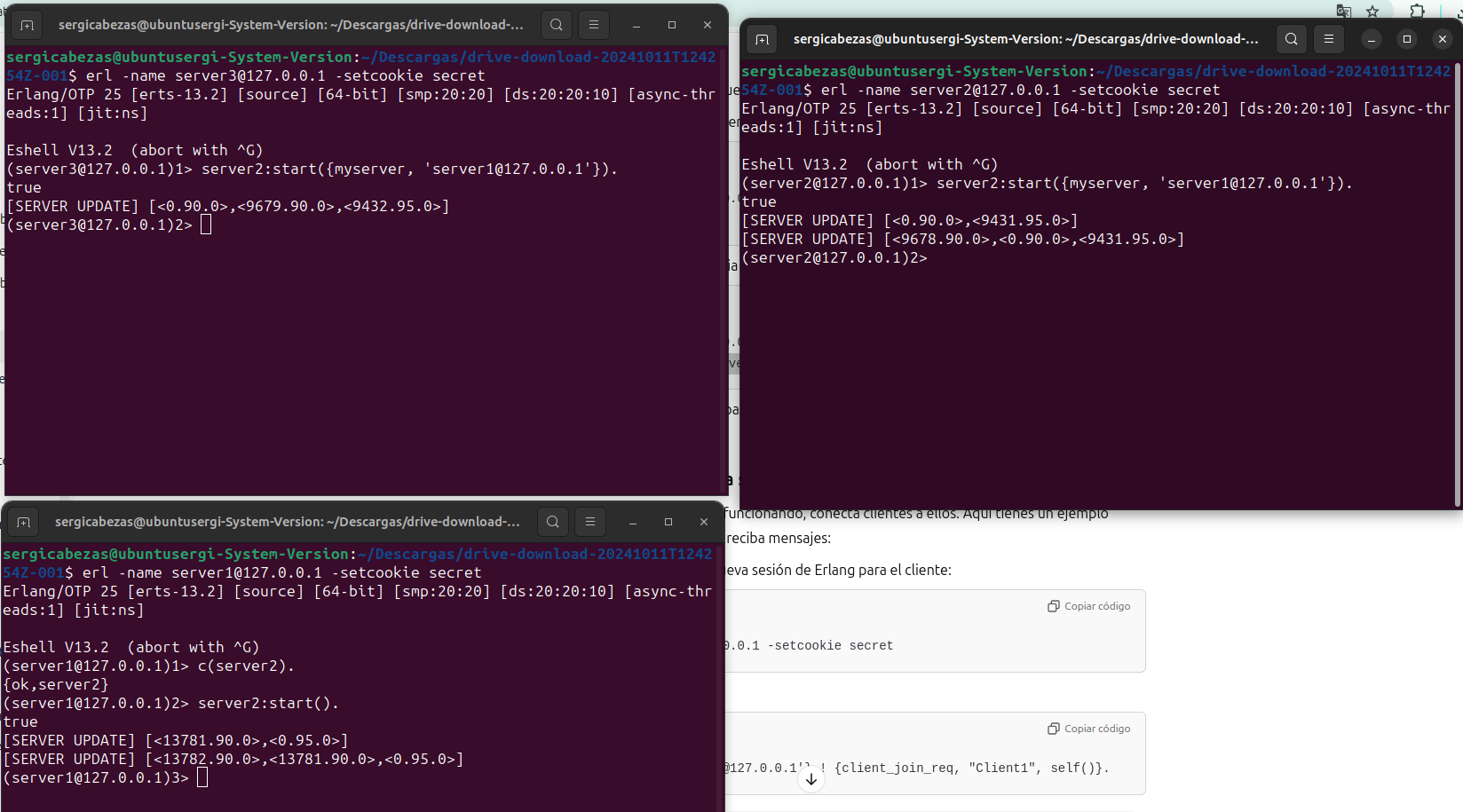
**vi) If a user joins or leaves the chat while the server is broadcasting a message, will he/she receive that message?**

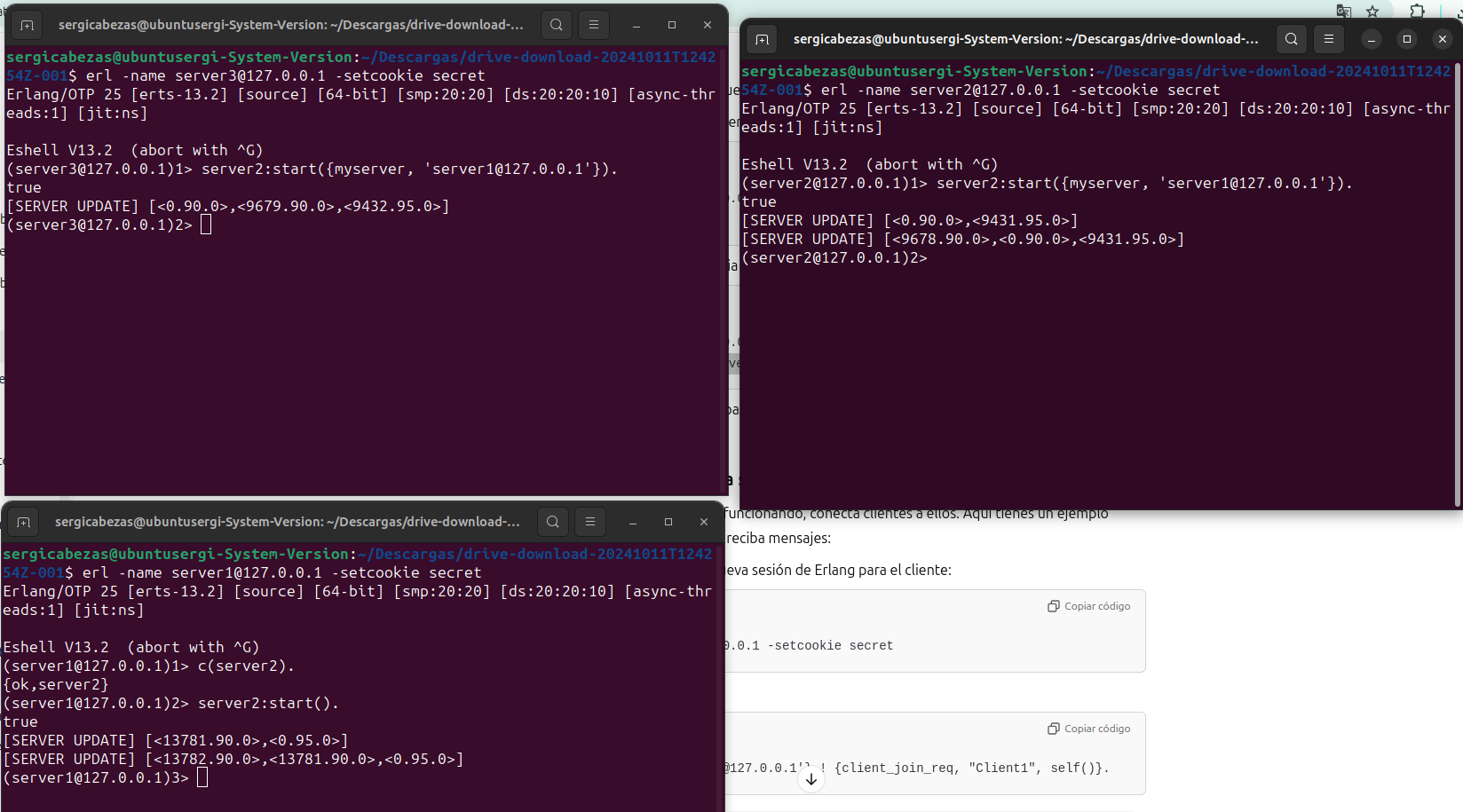
L'usuari no entrarà fins que el missatge hagi sigut retransmet, per tant, aquest nou usuari no podrà veure el missatge, si un usuari inicia la desconnexió després que el missatge s'hagi enviat el podrà rebre.

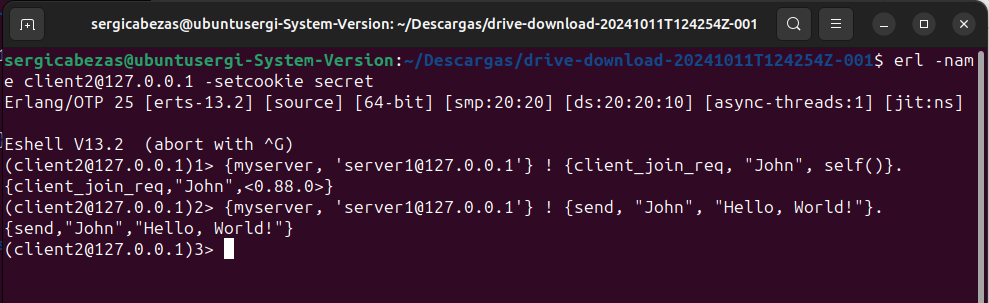
**PART 2**

**Experiments**







****

**Open Questions.**

**i) What happens if a server fails?**

Si el servidor falla, tots els clients connectats a ell perdran la connexió i no podran continuar comunicant-se entre si. Però a diferència d’abans, si aquest cau, no tot el sistema es veu interromput, ja que ara ens podem connectar a un altre servidor i podem seguir enviant missatges igualment.

**ii) Do your answers to previous questions iii, iv, and v still hold in this implementation?**

iii: En aquest cas sí. En Erlang, els missatges entre dos processos específics es continuen entregant en el mateix ordre en què s’han enviat.

iv: Els missatges enviats de manera concurrent per diferents clients poden arribar fora d’ordre, ja que el servidor els processa de manera seqüencial segons l’ordre de recepció, i aquest comportament no canvia.

v: Sí, és possible que un client rebi una resposta d’un altre client abans de rebre el missatge original d’un tercer client, ja que els missatges s'envien i es retransmeten de manera asíncrona pel servidor.

**iii) What happens if there are concurrent requests from servers to join or leave the system?**

Si diversos servidors sol·liciten de manera concurrent unir-se o abandonar el sistema, dependrà de com estigui dissenyat el servidor per gestionar-ho. Si no hi ha mecanismes de bloqueig podria provocar inconsistències. Per exemple, un client podria ser afegit o eliminat de la llista diverses vegades o cap. Per tant, es necessiten tècniques i processos de sincronització adequats per garantir que les operacions d’unió i sortida es processin de manera seqüencial i consistent.

**iv) What are the advantages and disadvantages of this implementation regarding the previous one?**

Avantatges:

* Simplicitat: La implementació és senzilla i fàcil d’entendre.
* Servidor únic: Al tindre un únic servidor es facilita la gestió de les connexions dels clients i la transmissió dels missatges.

Inconvenients:

* Escalabilitat: A mesura que augmenta el nombre de clients, el servidor únic es converteix en un coll d'ampolla provocant retards i dificultats per gestionar les peticions a temps.
* Tolerància a errors: Si el servidor falla, tot el sistema cau i cap client pot continuar comunicant-se.
* Concurrència: El servidor només pot gestionar una petició alhora. Les peticions concurrents dels clients, especialment en un sistema amb molts usuaris, poden causar retards o errors de processament.