BIFI/unizar and Kampal SL

Planning for AI bots in
Kampal Collective Intelligence App

What is collective intelligence?

- Trivial: average of estimates
- Weak: Is a group of persons more intelligent, that the most intelligent of the group?
  - (for a given problem. Think e.g. chess)
- Strong: Can a group of persons generate an idea, or find a solution to a problem, that can not be grasped/understood by one single person?
  - (given enough time and resources)

How to test it?

- Online tool.
- Enough number of persons. Order 100… 1000
- Monitorize activity
- Converge to a unique solution.

The four sins of collaboration tools, or the curse of social networks

- Noise
- Isolation Bubbles
- Conformity
- Leads and Follows

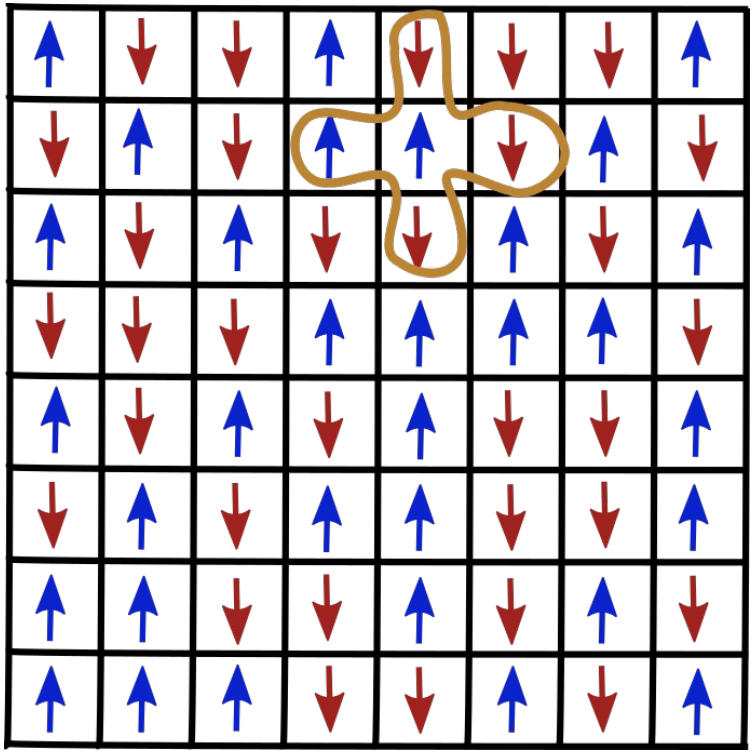
An example of complex scale-free network.

Martin Grandjean - Grandjean, Martin (2014). "La connaissance est un réseau". *Les Cahiers du Numérique* **10** (3): 37-54. DOI:10.3166/LCN.10.3.37-54.

All at the same time!

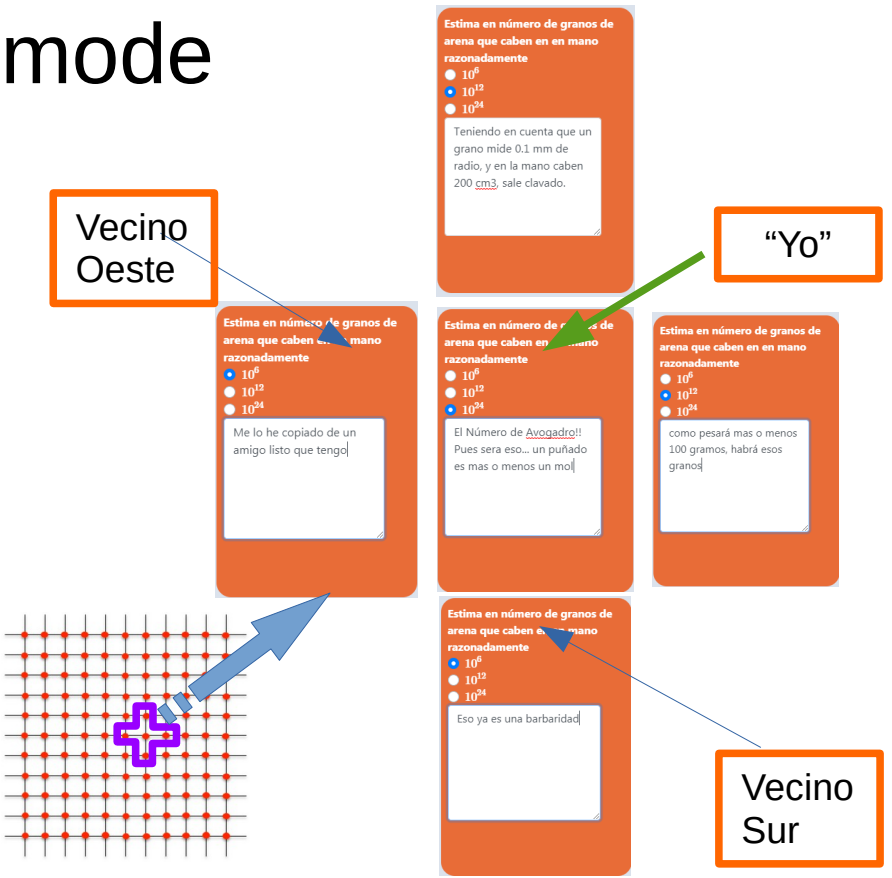
Do not use scale free networks

- Go the Ising way instead
  - No noise, no leaders
  - Definite rules for:
    - Copying items
    - Delete items
    - Long Range Exchange

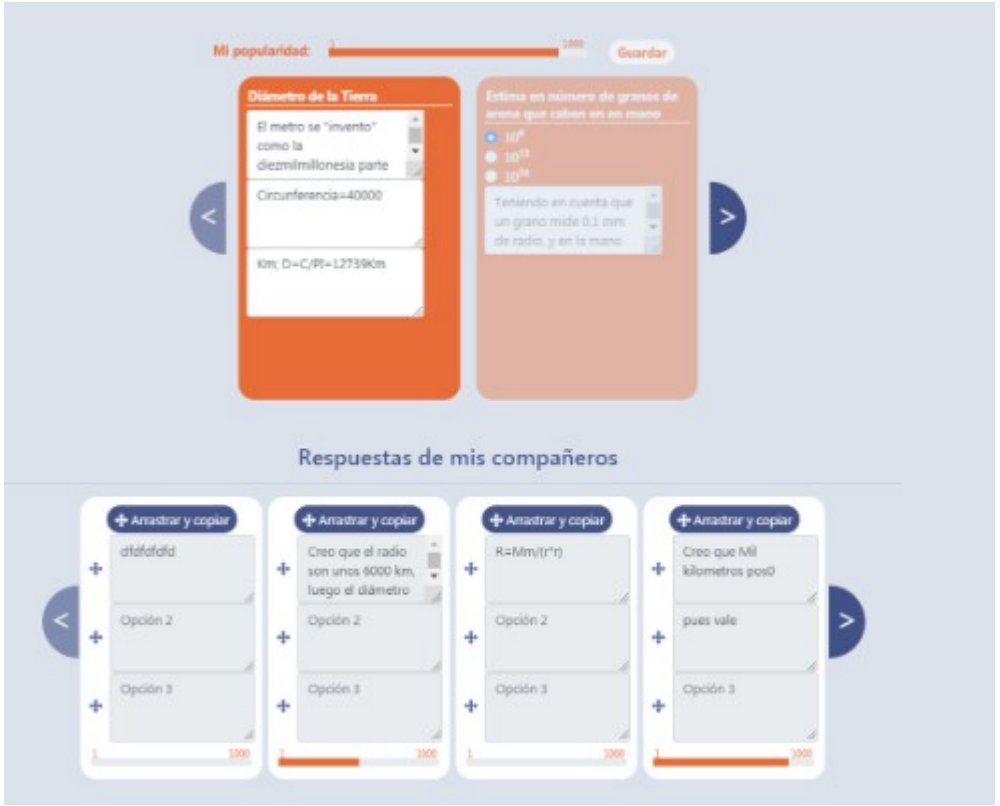


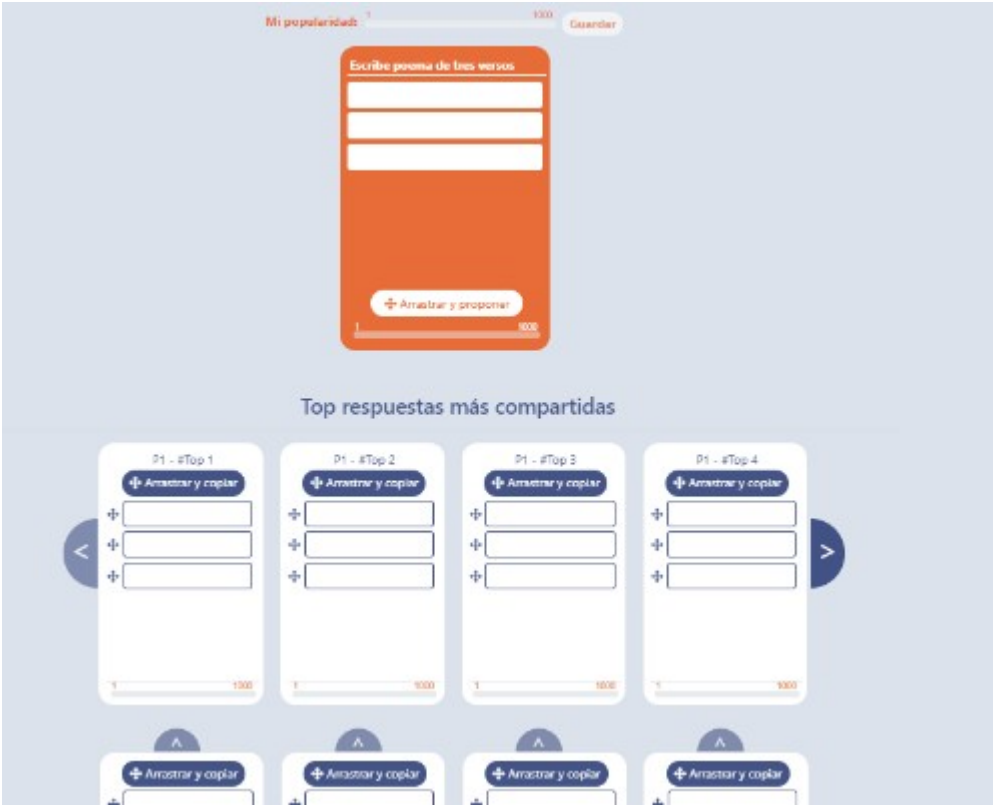
Collective Intelligence app

Neighborhood mode



(also with more compact UI)





See https://ic.kampal.com/info

So, now add bots. Why?

- Initial motivation:
  - Developers need debug, pressure the system for high computational load.
  - Psychologists need infiltrated agents: test the resilence of the system to coordinated action to induce a given answer
- New motivations:
  - Coordinate the actions of humans and machines
  - Just NPC: ¿Is a crowd of agents more intelligent than the most intelligent agent of the crowd?

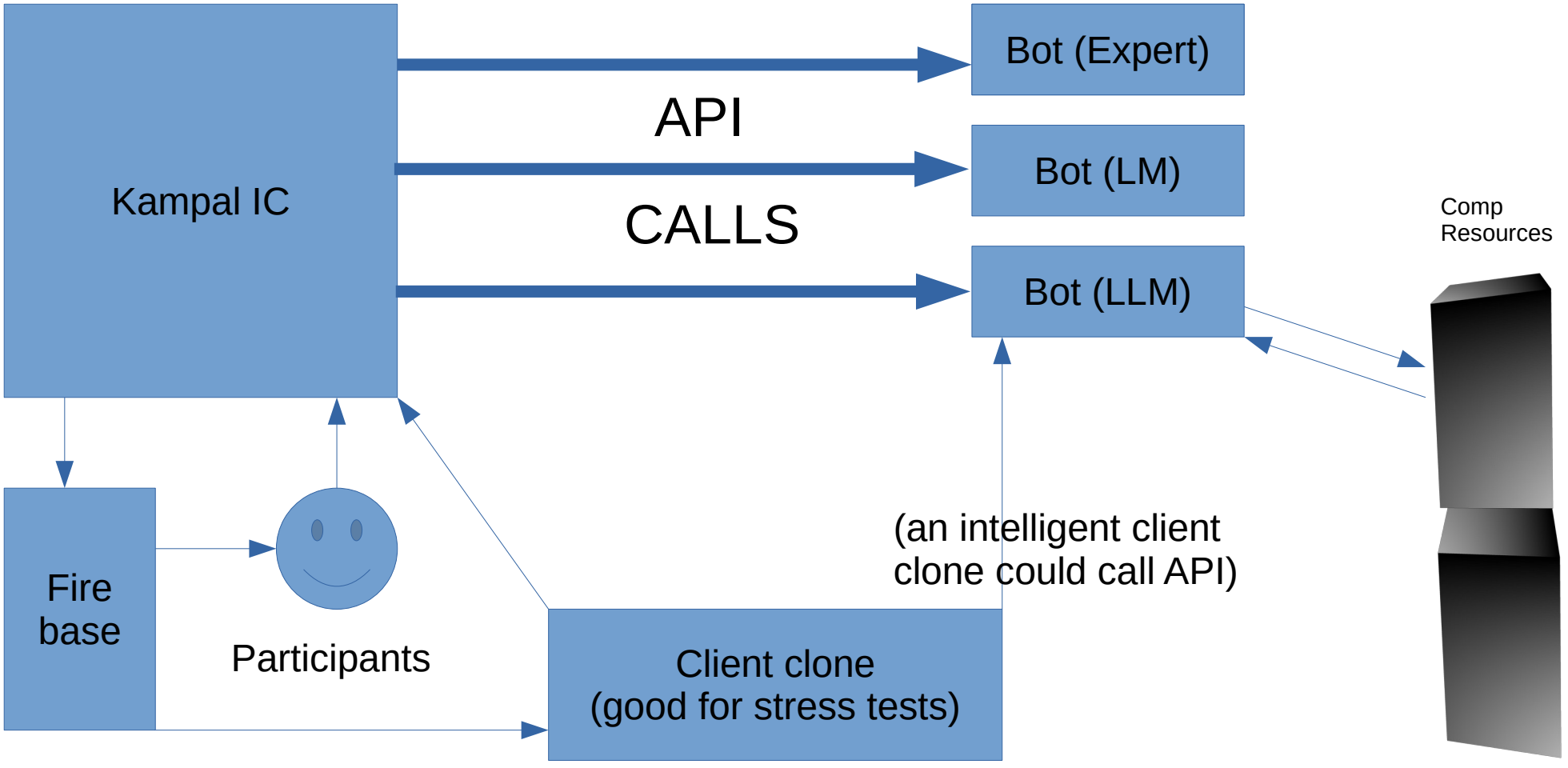
How?

- Clone the human client http:
  - Good for stress test
  - Bad for control, bad for profiles.
  - Good/Bad: It could bootstrap a complelety different project: automated navigation with AI
- Create an API to send commands to the AI session:
  - Bad for stress test
  - Good for fine control
  - Good to test multiple approaches to AI. Including classical expert systems.

Async or batch considerations

- We could get an answer too late, when the user has already been moved to other position, or neighboorhood has changed answers
- But it already happens with humans too! The user takes time to copy and edit an aswer, changes can happen. We always allow the human input, as it is consequence of real interactions, even if in the “past lightcone”

Graphically…



The payload

- Always some variation of json payload
  - {solution_id: “…”, project_message:”…”, question_message:”…”, current_answers: Array[…]}
  - Returns: change or array of changes, solution_id, timestamp
- Important: it is AI-agnostic. The API server is the one who knows what kind of bots are available and how to execute them, as well as sesion keeping etc.

Resources needed

- Non interactive for training, interactive for execution.
- Depends of course of bot type:
  - Bots that just choose an answer: can be any kind of string evaluation, without training. Or can be not very large LM, such as MarIA or any GPT-2, with just some fine tuning for training.
  - Bots that create new answers: most surely will be open sourced LLMs as LlaMa or similar, downsized for execution without GPU. Can benefit of training, or can work as one shot or few shots chat.
- Training always benefit of GPU in clusters. Execution, as it needs just human comparable speed, can be GPU or CPU based.

Thanks