

# PARTY HARD

**2015/2016**

**Gonzalo del Palacio Jiménez  
Kevin Garcia Santos**

# ESTRUCTURA

## Agentes :::

- Invitado
- PartyAgent
- Camarero

## Dominio :::

- Comensal
- Camarero

## Comportamientos :::

- registrar  
(invitados)
- saludar  
(invitados)
- comer (invitados)
- Servir (camareros)
- PartyAgentCore  
(partyAgent)

## Mesajería ::

- messageSystem  
: FIPA

# AGENTES

- **Invitado:** Se registra en la fiesta, saluda, come, y, una vez que está satisfecho (cuando le sirven 3 veces) se despide del anfitrión y se va.
- **Camarero:** Cuando el anfitrión le da permiso para servir la comida va sirviendo a los invitados de forma aleatoria y con intervalos aleatorios.
- **partyAgent: Tiene 2 “facetas”:**
  - a. **Anfitrión:** Saluda a los recién llegados, da la señal de servir la comida, come, se despide de los que se van y se encarga de “cerrar la fiesta”.
  - b. **controlador (interfaz):** Lleva un registro de los invitados registrados, de las veces que han comido, de si están satisfechos y de las veces que sirve cada camarero.

# DOMINIO

- Comensal: Una clase que permite almacenar para cada comensal su:
  - aid.
  - si está satisfecho.
  - veces servido.
- camarero: Una clase que permite almacenar para cada camarero su:
  - aid.
  - veces que ha servido.

# COMPORTAMIENTO CAMARERO

- **servir:** TickerBehaviour

- Cuando reciben un mensaje por parte del anfitrión indicando que se han marchado todos los invitados, los camareros se van.
- Si en un random  $\geq 25$ 
  - Se sirve a un invitado aleatorio y se notifica al controlador que el camarero x ha servido.
- sino
  - Se sirve al anfitrión y se notifica al controlador que el camarero x ha servido.

# COMPORTAMIENTOS INVITADO (1/3)

- **registrar:** OneShotBehaviour
  - Inserta al nuevo invitado registrado en el grupo de invitados.
  - Saluda al resto de invitados en la fiesta.
  - Saluda al anfitrión.
  - Añade al invitado el comportamiento de saludar.

# COMPORTAMIENTOS INVITADO (2/3)

- **saludar:** CyclicBehaviour

- Si un invitado nuevo [N] (que no estaba antes de que se registrase este invitado [A]) saluda a todos, el invitado actual [A] le responderá el saludo .
- Si un invitado recibe la notificación por parte del anfitrión de que es hora de comer, se le asigna el comportamiento Comer.

# COMPORTAMIENTOS INVITADO (3/3)

- **comer:** `CyclicBehaviour`

- Si un invitado es servido por un camarero, se notifica al controlador para que actualice en la interfaz las veces que le han servido.
- Si un invitado recibe la notificación por parte del controlador de qué está satisfecho, este primero, se despide del anfitrión.
- Si un invitado recibe el saludo de despedida por parte de anfitrión, se va de la fiesta.



# COMPORTAMIENTO PARTYAGENT

- **PartyAgentCore:** `CyclicBehaviour`

- a. Anfitrión:

- Si recibe un saludo por parte de un invitado nuevo, le devuelve el saludo.
  - Si se han registrado todos los invitados les indica de que es hora de comer y a los camareros de que es hora de servir.
- Recibe mensajes de que le han servido la comida.
- Si un invitado se despide de anfitrión, este le devuelve la despedida.

- b. Controlador (Interfaz):

- visualiza los invitados activos en cada momento.
- visualiza las veces que come cada invitado.
- visualiza si un invitado está satisfecho.
- visualiza las veces que sirve cada camarero.

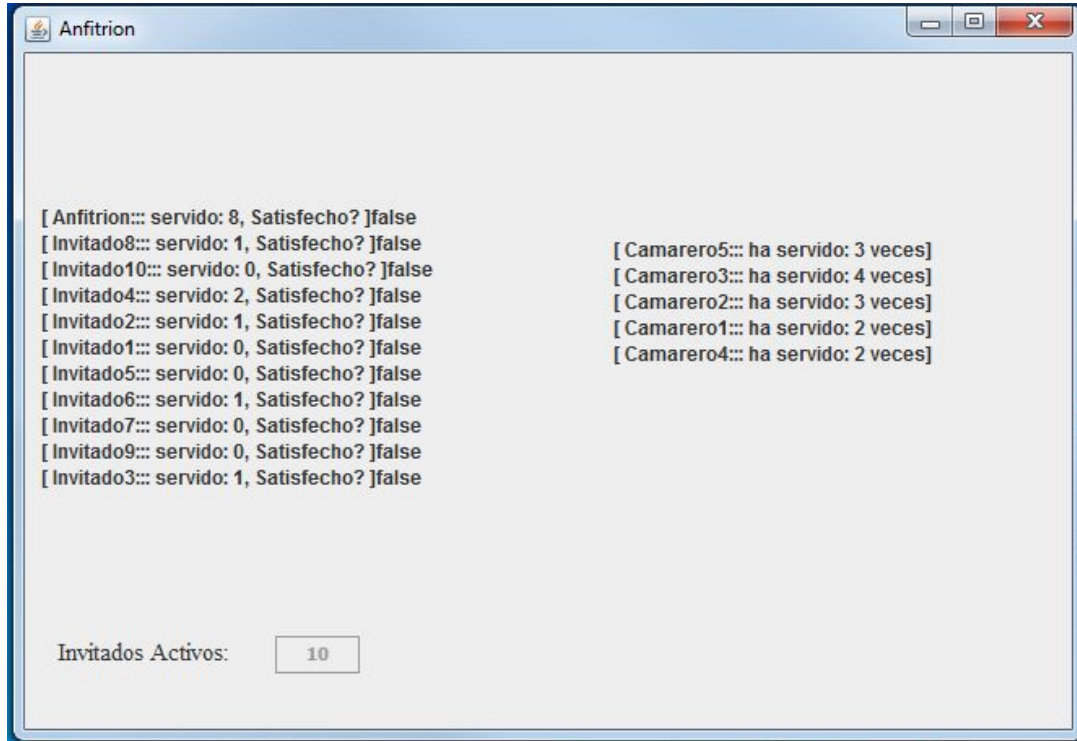
# MENSAJERÍA

- `messageSystem: FIPA`
  - `sendToGroup :: Envía un mensaje (m) a todos los miembros del grupo (x).`
  - `sendToGroup_Random :: Envía un mensaje (m) a a un miembro aleatorio del grupo (x).`
  - `sendTo :: Envía un mensaje (m) al agente(x).`

# TABLA DE MENSAJES

MSJ	invitado	Anfitrión	Camarero	Controlador
saludar_todos	envía/recibe	recibe		
saludar_uno	envía/recibe	envía/recibe		
ya	recibe	envía	recibe	
servir	recibe	recibe	envía	
me_han_servido	envía			recibe
ha_servido			envía	recibe
satisfecho	recibe			envía
despedirse	envía	recibe		
ok	recibe	envía		
terminar		envía	recibe	

# PROGRAMA EN EJECUCIÓN



Comentarios de la ejecución.