Globulation 2

Free software RTS game with a new take on micro-management

http://www.globulation2.org

Stéphane Magnenat

with help and feedback from the community

February 23, 2008

Acknowledgements

Thanks to everyone who contributed time and resources to Globulation 2. This game would not be what it is without your support.

Contributors are listed in the AUTHORS file in the Globulation 2 distribution.

Outline

Section

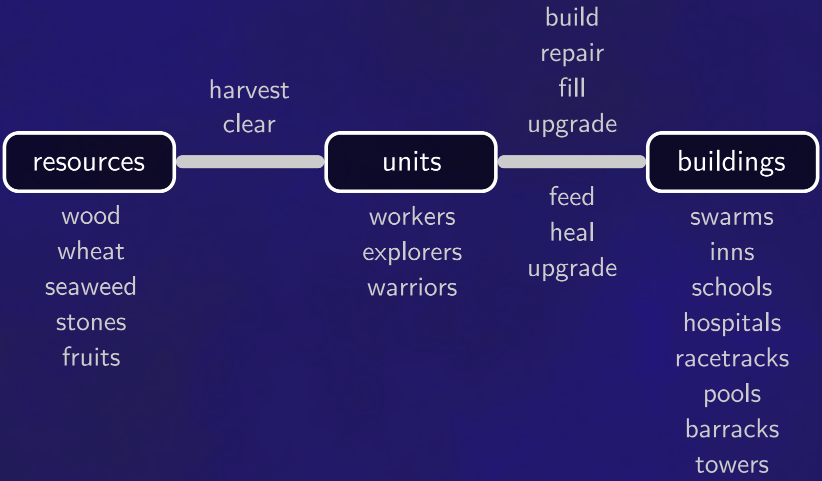
Founding Principles

Rationale behind the Globulation adventure

A strategy game should focus on strategy, not on micro-management.

Inspiration

The Ecology of Globulation



Globulation 1

1999–2000, 20000 lines of Think Pascal, Mac OS, single player
Units have their own lives, they individually upgrade, work, eat...

Player

chooses ratio of

- buildings
- units

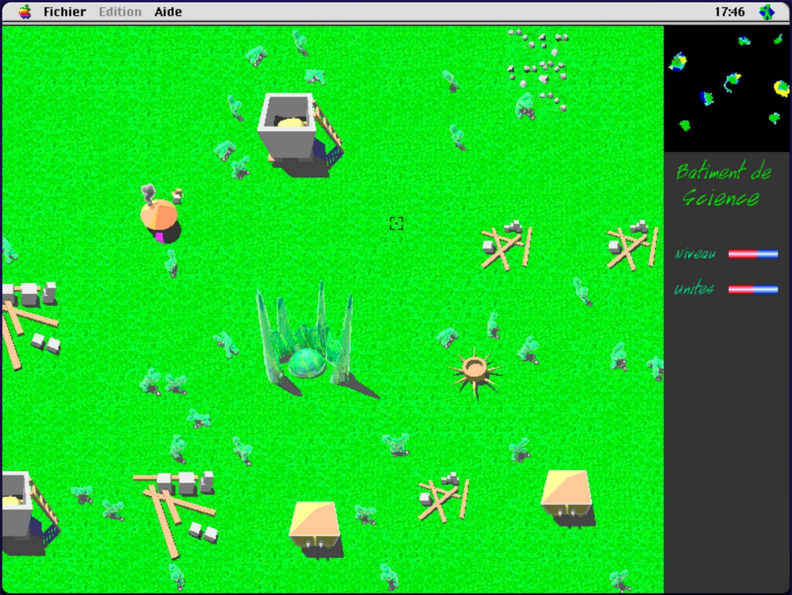
Game

manages units, which

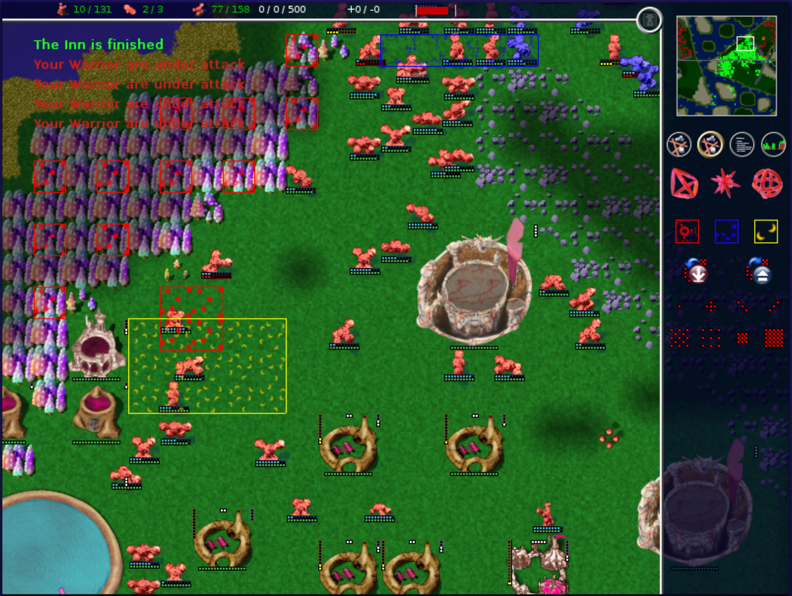
- move randomly
- place buildings randomly
- build buildings randomly
- find resources randomly
- go to resources using pheromones

The resulting gameplay is fun to watch, but boring to play.

Globulation 1



Globulation 2



The Inn is finished

Your Warrior are under attack

Your Warrior are under attack

Your Warrior are under attack

Your Warrior are under attack

Globulation 2

2001–2008, 100000 lines of C++, cross-platform, multiplayer
Although units retain their own lives, the player has a wider range of actions.

Player

- places buildings
- upgrades/repairs
- places flags
- sets the number of units
- specifies areas

Game

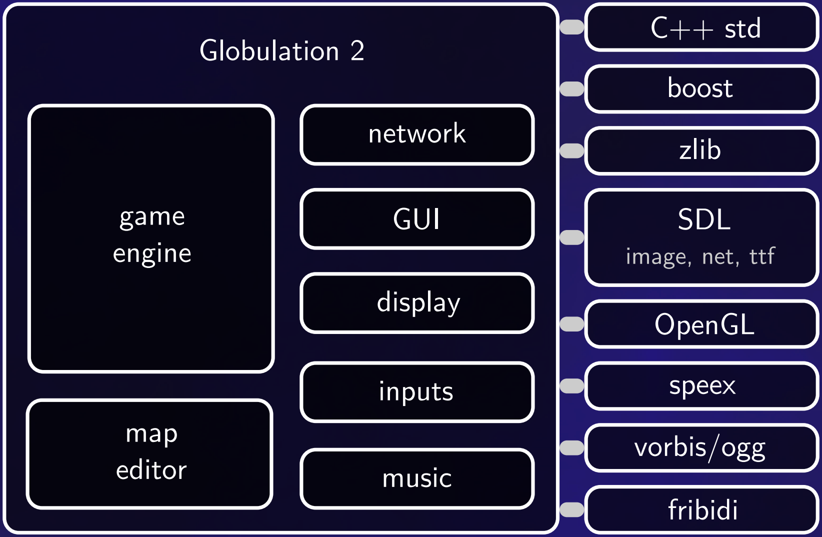
- assigns units to buildings and flags
- manages units food, health, and upgrades
- provides pathfinding

The resulting gameplay is innovative, fun, and extensible.

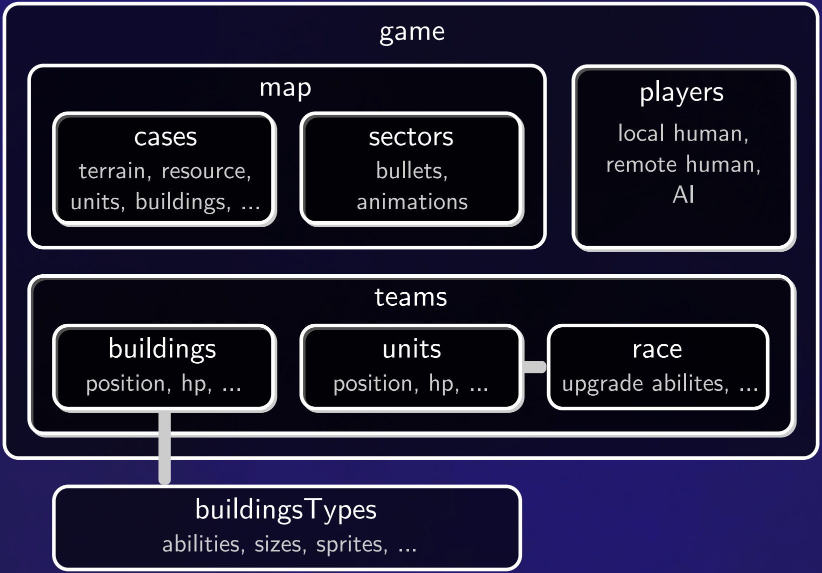
Section

Architecture

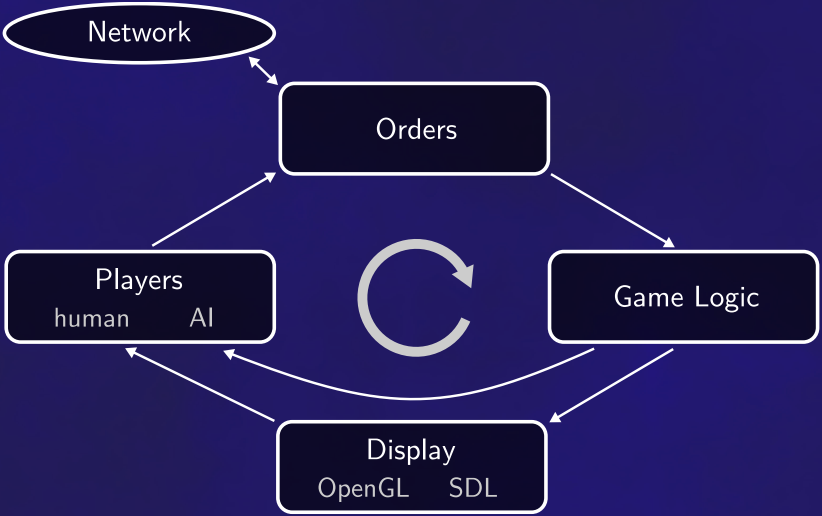
Overall Structure



Game Engine Structure



Information Flow



Section

Network Model

Network Model

Synchronous game engine

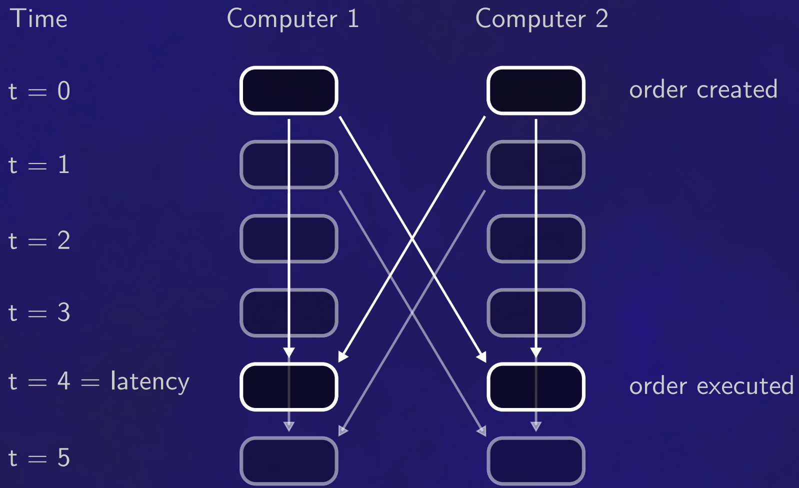
Features

- TCP, meta-server based (originally UDP, P2P)
- meta-server initiates connections, routes data
- players only exchange orders
- smallest possible bandwidth
- small, uniform latency
- complete game state checksummed
- state modification cheating impossible

Drawbacks

- code execution must be predictable (no float, only stable_sort, care with sets, . . .)
- cannot prevent view cheating

Network Model



Section

Pathfinding and Task Allocation

Pathfinding

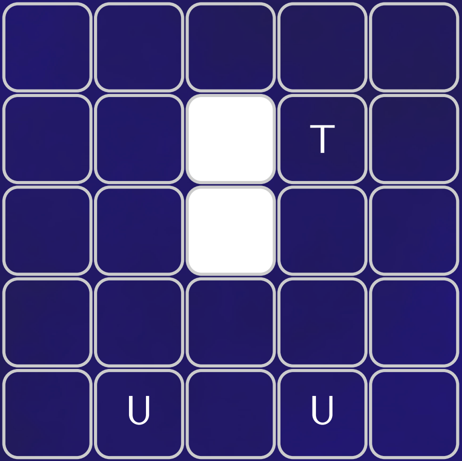
In the game

- linked to targets: buildings, resources, areas
- used by units
- created/updated on demand
- locally overridden upon congestion
- takes a large amount of CPU time
- took a substantial amout of development time
- must be perfect, otherwise it kills your game (unless you are Blizzard)

As an algorithm

- gradient to target
- creation using gradient propagation (NF1, grassfire)
- used as gradient ascent
- complete exploration of all accessible map parts
- $\mathcal{O}$(targets count $\times$ map width $\times$ map height)

Pathfinding



Pathfinding

		254	254	254
		⬜	T	254
		⬜	254	254
	U		U	

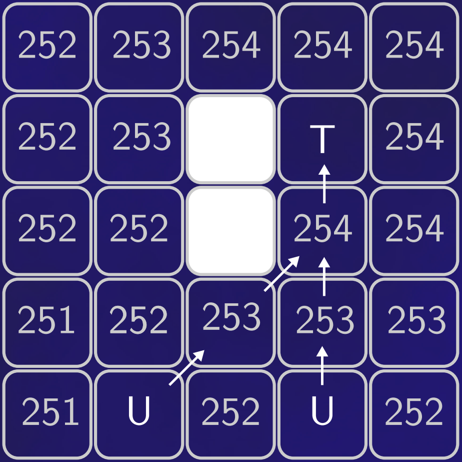
Pathfinding

	253	254	254	254
	253	⬜	T	254
		⬜	254	254
		253	253	253
	U		U	

Pathfinding

252	253	254	254	254
252	253	⬜	T	254
252	252	⬜	254	254
	252	253	253	253
	U	252	U	252

Pathfinding



252	253	254	254	254
252	253	🟦	T	254
252	252	🟦	254	254
251	252	253	253	253
251	U	252	U	252

Task Allocation

- market based approach
- free units subscribe to lists
- demanding buildings subscribe to lists
- priority per building type; inns first, higher level first
- greedy allocation, one unit per building per allocation round

Section

Community

Interfaces with the Community

With the binary

- map and campaign editor
- translations
- testing and gameplay tuning
- virtual filesystem: graphics, music
- documentation

With the sources

- coding
- documentation

On the web

- wiki based homepage
- IRC, YOG
- mailing lists: mostly developers
- forum: mostly players
- bug tracker
- mercurial repository

The Globulation 2 community needs you!

Scripting

Story 1

```
show("build 10 schools")
wait(10)
hide
wait(School(0, 1) > 9)
win(0)
loose(1)
```

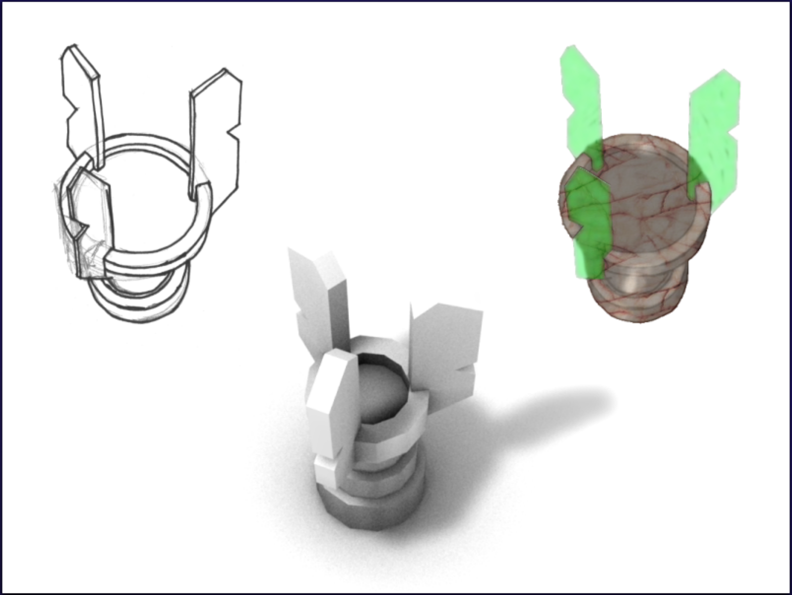
Story 2

```
wait(10)

# lvl. 1 schools of p. 1
wait(School(1, 1) > 9)
win(1)
loose(0)
```

- rudimentary, not generic, but easy to use
- multithreaded, safe, synchronous, serializable
- next generation version in the pipeline

Graphism Creation



Contributors Capture and Storage

- open source developers are highly volatile resources
- especially artists
- complexity is both boon and bane
- people come, implement, disappear; don't document much
- people like to reinvent better wheels each time
- must maintain balance between guiding new developers and letting them express their visions

Section

Conclusion and Future

Conclusion and Future

Current situation

- code base is stable
- community is stabilizing
- core engine scales well
- gameplay is innovative and promising

In the future

- tune gameplay
- improve campaigns
- improve user friendliness
- further reduce micro-management
- add gameplay elements
- if enough demand and artwork, 3D graphics

Take Home Message

Gameplay, atmosphere, and artwork are critical for success

Take Home Message

Gameplay, atmosphere, and artwork are critical for success

Join the Globulation adventure and have fun!

Time for Questions

Feel free to express yourself.

Copyrights