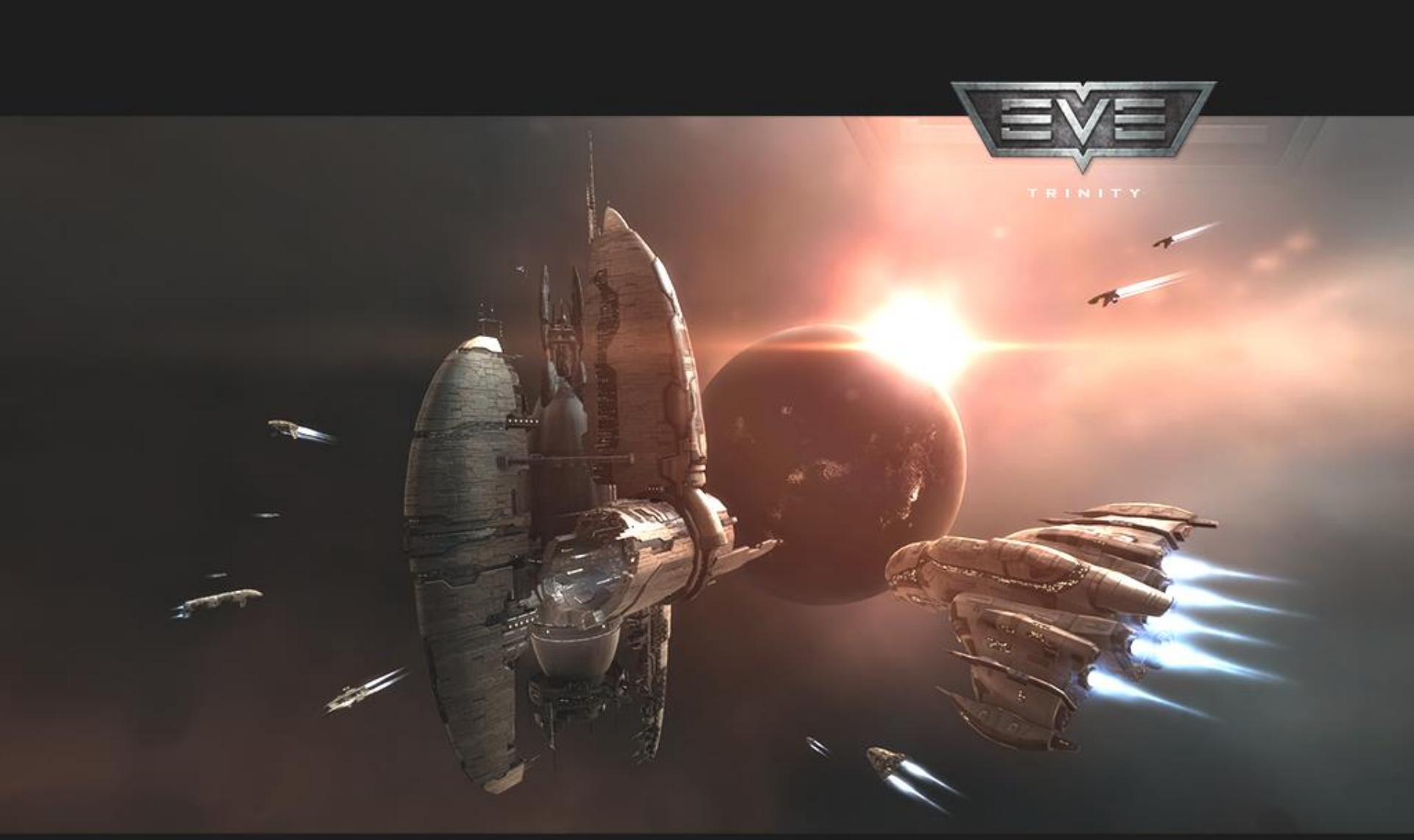
- **The Server Technology of EVE Online: How to Cope With 300,000 Players on One Server**
  Speaker: Halldor Fannar Guðjónsson (CTO, CCP)
  Date/Time: Wednesday (September 17, 2008) 4:30pm — 5:30pm
  Location (room): Room 4
  Track: Online Games - Technology and Services
  Format: *60-minute* Lecture
  Experience Level: Intermediate

  Session Description
  Most MMOGs separate their players onto multiple servers in order to scale; these servers are often called shards. The shards are essentially instances of the game world and the player base is thus broken into segments, based on which shard (server) they are playing on. EVE Online takes a different approach: the technology and the game is developed around the goal of one unified world. All the players log onto the same server and play in the same world. This session will explore the technology developed by CCP to achieve this goal and discuss the technology choices made by CCP, both on the hardware and software side.

  Idea Takeaway
  Learn about the problems that game developers typically face when scaling an MMOG server. A number of solutions presented – all in the context of a successful MMOG.

  Intended Audience
  Programmers and game designers who need to network a large world of players. People curious to see how you can face a daunting problem and find a successful solution.



The Server Technology of EVE Online:

How to cope with 300,000 players on one server

About EVE

- One world, one server cluster
- Launched in May 2003, has been steadily growing ever since
- Currently at over 300,000 players (~250,000 paying subscribers, trials account for rest)
- Peak Concurrent Users (PCU) record at 42,711
- That's over 13 miles (21 km) of people

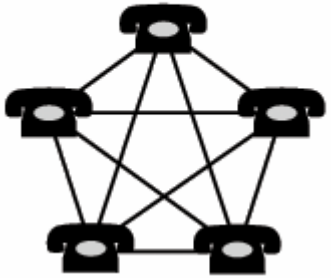
The Network Effect

Network effect

The value of a service to a potential customer is dependent on the number of customers already using that service

Metcalfe's law

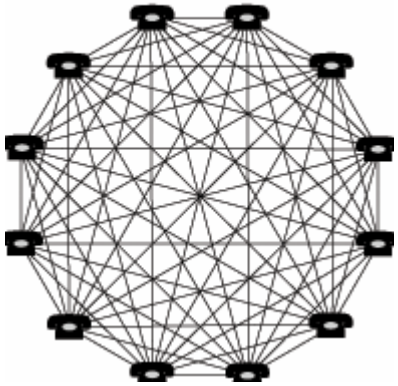
Value of a network equals approximately the square of the number of users in the system



Ergo

EVE Online is inherently a better game with **250.000** subscribers than **50.000** subscribers

Size Matters!



About myself

ATARI

MIDWAY

Lightspeed Games

MAXIS

EA

CCP

For more details

- E-ON magazine

There is soooo much to talk about

- Won’t be able to cover everything
- This isn’t simple stuff, I have taken the liberty of leaving out some complex details
- The above prevents death from tedium, brain explosions and allows us to cover more ground
- Don’t worry, this lecture will still be very technical
- We will mix it up with some demos and videos

The main reasons for EVE's scalability

- Database system
- Stackless Python
- Destiny and space partitioning
- Hardware configuration

EVE is built from the database and up

- Efficient management of data in an un-sharded MMOG is crucial
- Python allows for Lisp-like data handling; native support for lists, dictionaries, etc.
- EVE database is currently at 1.3 terabytes
- Database transactions are usually around 1500 per second, reaching 2000 per second at peak hours
- ‘invItems’ table receives over 9 million insertions in a day (not counting update and select statements)

EVE demands an enterprise database solution

- Microsoft SQL 2005 x64 (enterprise edition)
- IBM X-series 3950 brick server
- 64 GB RAM
- 4 x CPU, each dual-core at 3.5 GHz
- 2x 128 GB RamSan 400 (solid-state drive)

EVE demands enterprise level database engineers

- Tables and procedures are carefully designed
- Constant monitoring (through DB traces, health checks on DB server) to watch for bottlenecks and optimization opportunities
- Optimizations sometimes trickle all the way up to game design
- Recently created robust versioning system for DB content, you will eventually need this [link]

A better way to program

What we wanted

A scripting language

A procedurally synchronous environment

We tried many things

We almost got it working with an implementation of JavaScript and NT Fibers

Python 1.5.2

As soon as we found Stackless Python early summer 2000, we knew we had a winner

python™

Procedural Synchronous
vs
Callback driven Asynchronous

What is Stackless Python?

An alternate version of Python

- Forked from the Python code
- Contains additional features
- Completely backwards compatible

Why Stackless?

- Lightweight threading (tasklets)
- Allows you to write more readable code with less boilerplate
- Allows switching of running code regardless of what is on the call stack (third party libraries, C/C++, etc)
- Dynamic reload of modified code (in-house tech)
- It’s just so damn cool

Co-operative Tasklets (live demo)

```
def func( string ):
    ...     for i in xrange( 10000 ):
    ...         print string
    ...         stackless.schedule()


import stackless
t = stackless.tasklet( func )
s = stackless.tasklet( func )
t( ‘Obama’ )
s( ‘McCain’ )
```

Destiny

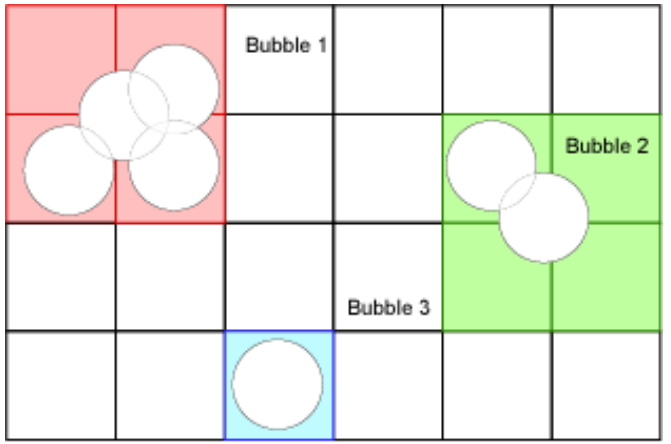
Divide and conquer	Space Partitioning dynamically calculates the causality bubble for each ship in a fast way
Deterministic	Allows state to be evaluated by the EVE client, knowing only initial state and time
Differential Equations	Ships are essentially balls rolling around in vector fields

Partitioning space into a hierarchy of boxes

Binary space partitioning in three dimensions. Each sphere belongs to only one box – the smallest box that is able to contain the spere.

$S_{t+1} \leftarrow f(T_t(S_t))$ becomes:

$$S_{t+1} = \{S^0_{t+1}, \ldots, S^M_{t+1}\} \leftarrow f(S^0_t) \otimes \ldots \otimes f(S^N_t)$$

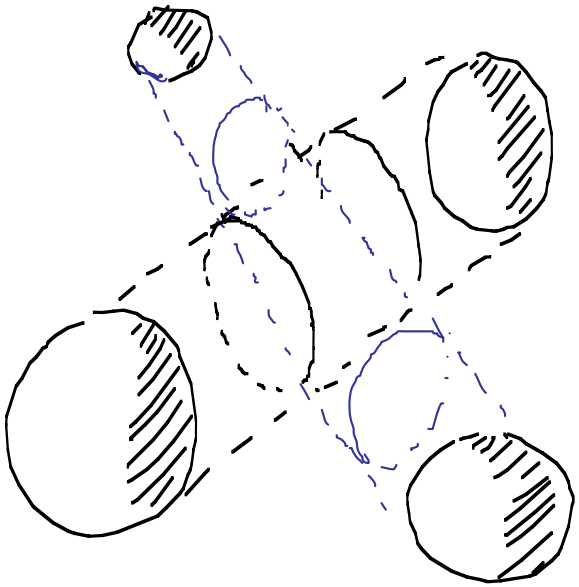


It is not as simple as the last slide implied

The time-extrusion of a sphere determines what space partitions it can interact with (causality bubble)

Collision is defined as the intersection of two time-extruded spheres (cylinders with dome shaped end-caps)

Collision response is highly elastic (conservation of momentum)



You have now seen Destiny

$$m\frac{d^2X}{dt^2} = F(X;t)$$

$$m\frac{d^2X}{dt^2} = a - k\frac{dX}{dt}$$

$$v(\infty) = \frac{a}{k}$$

$$t_1 = \frac{kD}{a}$$

$$a_x^2 + a_y^2 + a_z^2 = b_x^2 + b_y^2 + b_z^2$$

$$\frac{\partial S}{\partial t} = MS$$

$$k(x_2 - x_1) - mv_{1,x} = b_x t_1$$

$$k(y_2 - y_1) - mv_{1,y} = b_y t_1$$

$$k(z_2 - z_1) - mv_{1,z} = b_z t_1$$

$$\mathbf{F}_{i,collision} = \sum_{\substack{nearby \\ objects}} \nabla e^{-|\mathbf{R}-\mathbf{R}_i|^2/r_{max}^2}$$

$$D = X(t_1) + \frac{mV(t_1)}{k}$$

$$x(t) = x_0 + \frac{a_x t}{k} - \frac{m(a_x - v_{0,x}k)}{k^2}(1 - e^{-kt/m})$$

$$d = x(\infty) - x_0 = \frac{mv_0}{k}$$

$$X(0) = x_0 \quad and \quad \left.\frac{dX}{dt}\right|_{t=0} = v_0$$

$$V(t) = \frac{a}{k} - \frac{a - v_0 k}{k} e^{-kt/m}$$

$$X(t) = x_0 + \frac{at}{k} - \frac{m(a - v_0 k)}{k^2}(1 - e^{-kt/m})$$

$$\frac{S(t+\Delta t) - S(t)}{\Delta t} \approx MS(t)$$

$$t_1 = \frac{\sqrt{(k(x_2 - x_1) - mv_{1,x})^2 + (k(y_2 - y_1) - mv_{1,y})^2 + (k(z_2 - z_1) - mv_{1,z})^2}}{\|a\|}$$

$$\mathbf{F}_{i,goto} = -v_{i,\max}\begin{pmatrix} \tanh(p_x - x_i) \\ \tanh(p_y - y_i) \\ \tanh(p_z - y_i) \end{pmatrix}$$

$$\frac{\partial S}{\partial t} = F(S;t)$$

$$m_i\partial_t^2\mathbf{R}_i + \mathbf{F}_{i,viscous} + \mathbf{F}_{i,collision} + \mathbf{F}_{i,goto} + \mathbf{F}_{i,follow,j} = 0$$

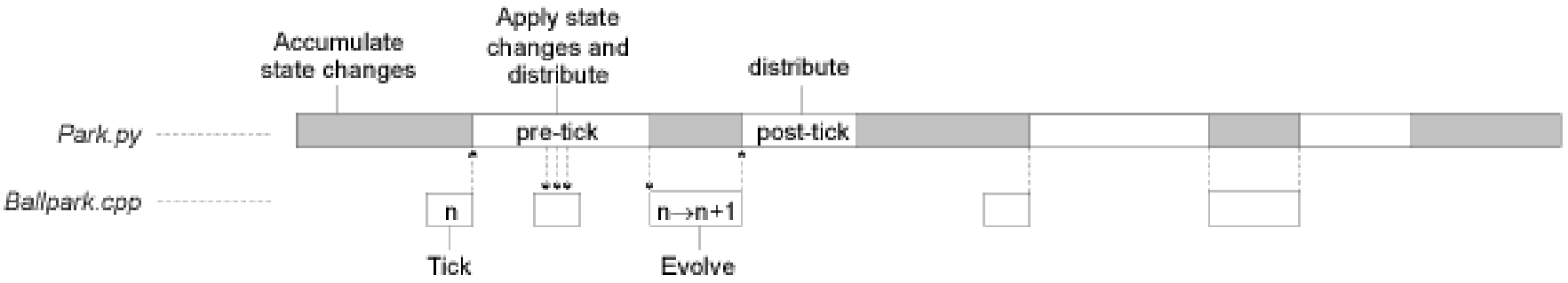
$$m_i\partial_t^2\mathbf{R}_i + k_i\partial_t\mathbf{R}_i + \sum_{\substack{nearby \\ objects}} \nabla e^{-|\mathbf{R}-\mathbf{R}_i|^2/r_{max}^2} - v_{i,\max}\begin{pmatrix} \tanh(p_x - x_i) \\ \tanh(p_y - y_i) \\ \tanh(p_z - y_i) \end{pmatrix} - v_{i,\max}\tanh\left[|\mathbf{R}_j - \mathbf{R}_i| - d_i(r_i + r_j)\right]\begin{pmatrix} \tanh(p_x - x_i) \\ \tanh(p_y - y_i) \\ \tanh(p_z - y_i) \end{pmatrix} = 0$$

QED

Evolving time

The server solves the system of equations for an entire “system”. The client only solves the equations for the causality bubble that it belongs to.

Server regularly sends updates to the initial conditions for a time step. The client can work its way backwards to adjust for changes in initial conditions and derive how they affect current state.



Two things now become obvious

- A client scales based on the population in a causality bubble
- A server scales based on the population in the “system” that we need to solve the differential equations for
- How are these concepts defined in EVE?
- Answers after this break!

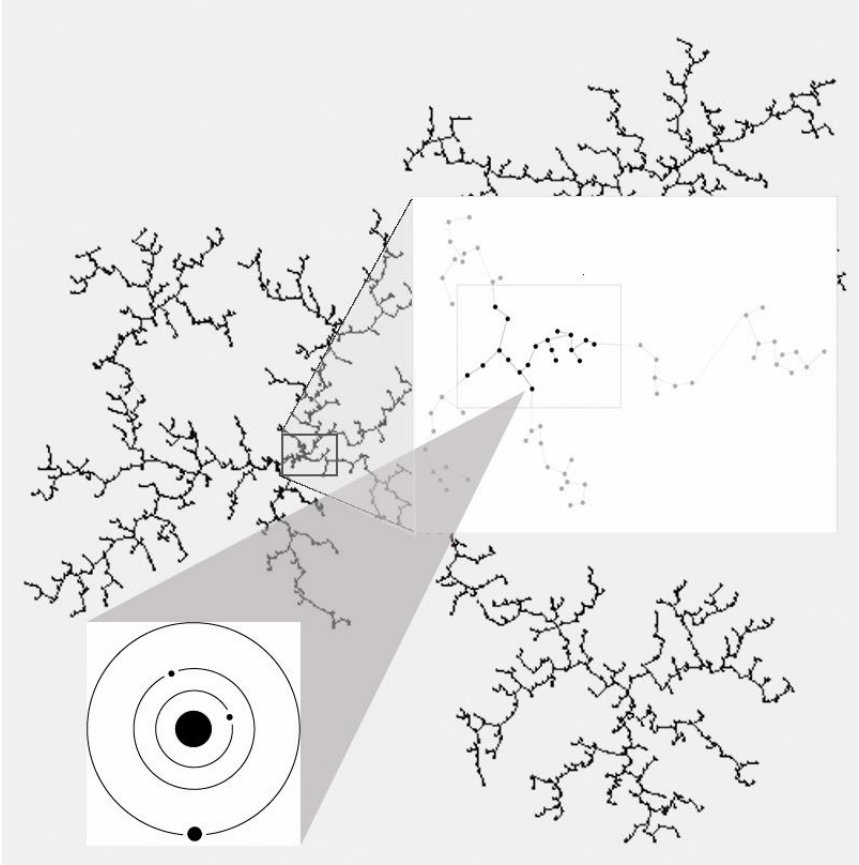
Movie time

The universe we wanted to create

Solar systems typically consists of a dozen of planets, surrounded by moons and asteroid belts

Routes connect them into constellations that themselves form conglomerates called regions

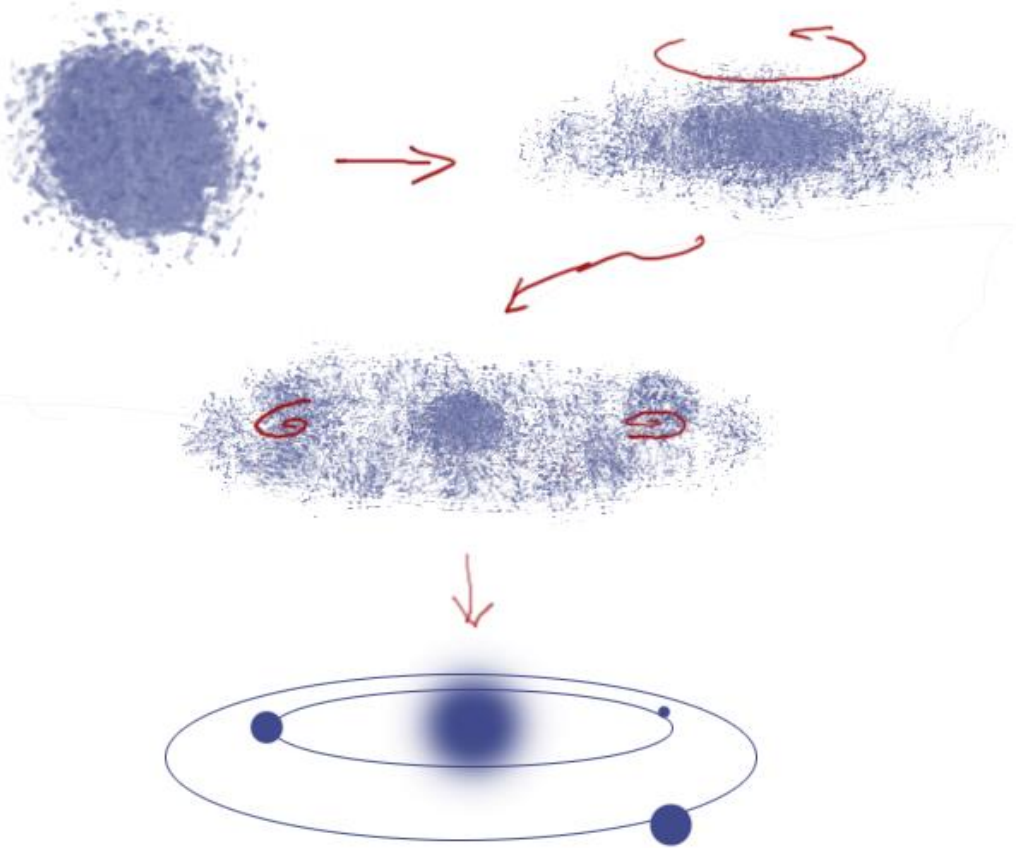
EVE consists of over 5000 solar systems



Cooking up solar systems

EVE solar systems are created using a **disc accretion model**

Gives convincing enough results and is possible to calculate in less than 24 hours

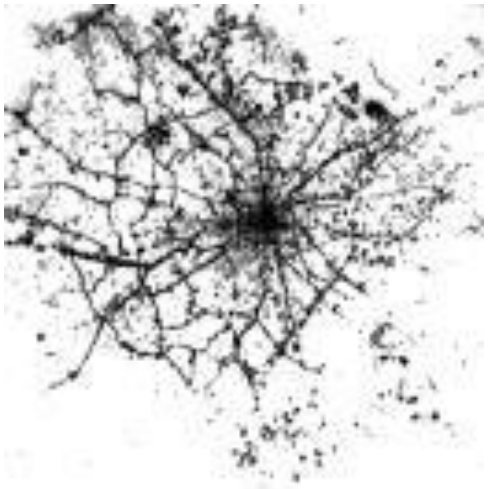


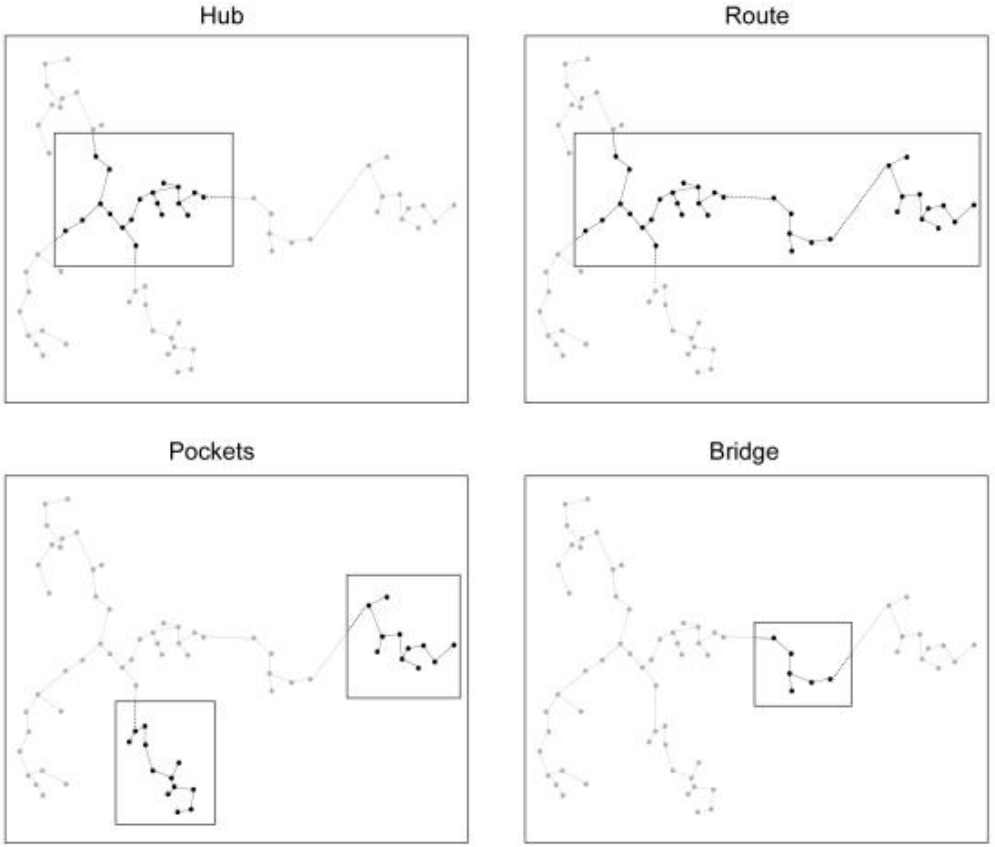
We wanted an interesting route system

Goals:

- Settlements
- Strategic
- Hierarchical

=> **Diffusion Limited Aggregation (DLA)**

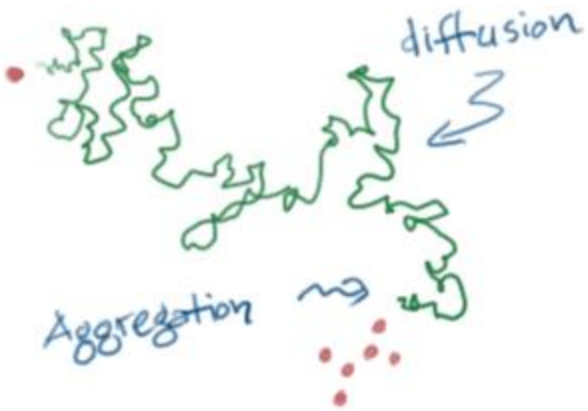




Generating the route system

Diffusion Limited Aggregation (DLA)

Extended to multi-seed and three dimensions



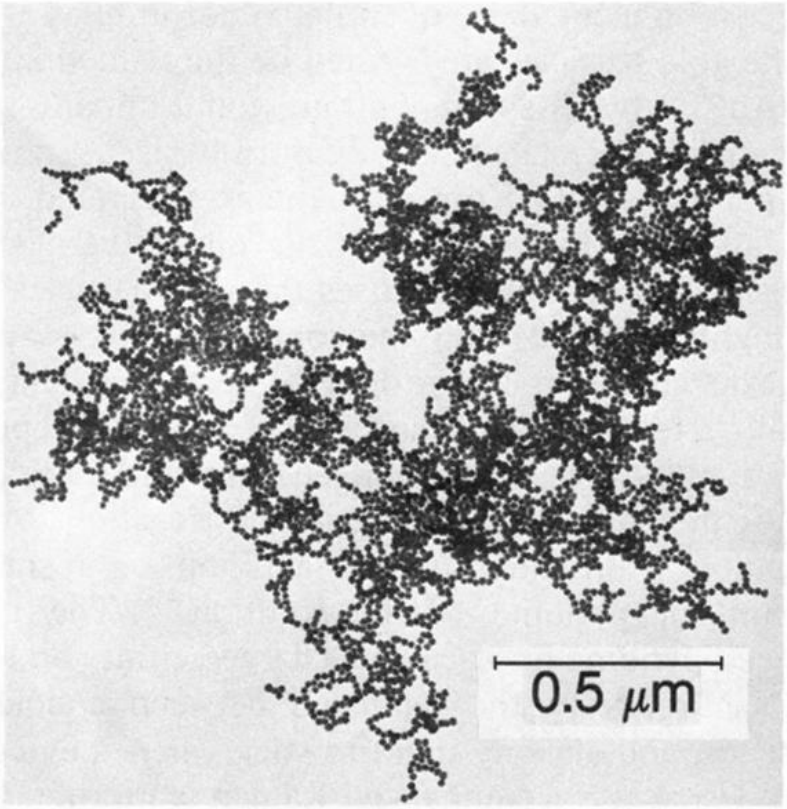
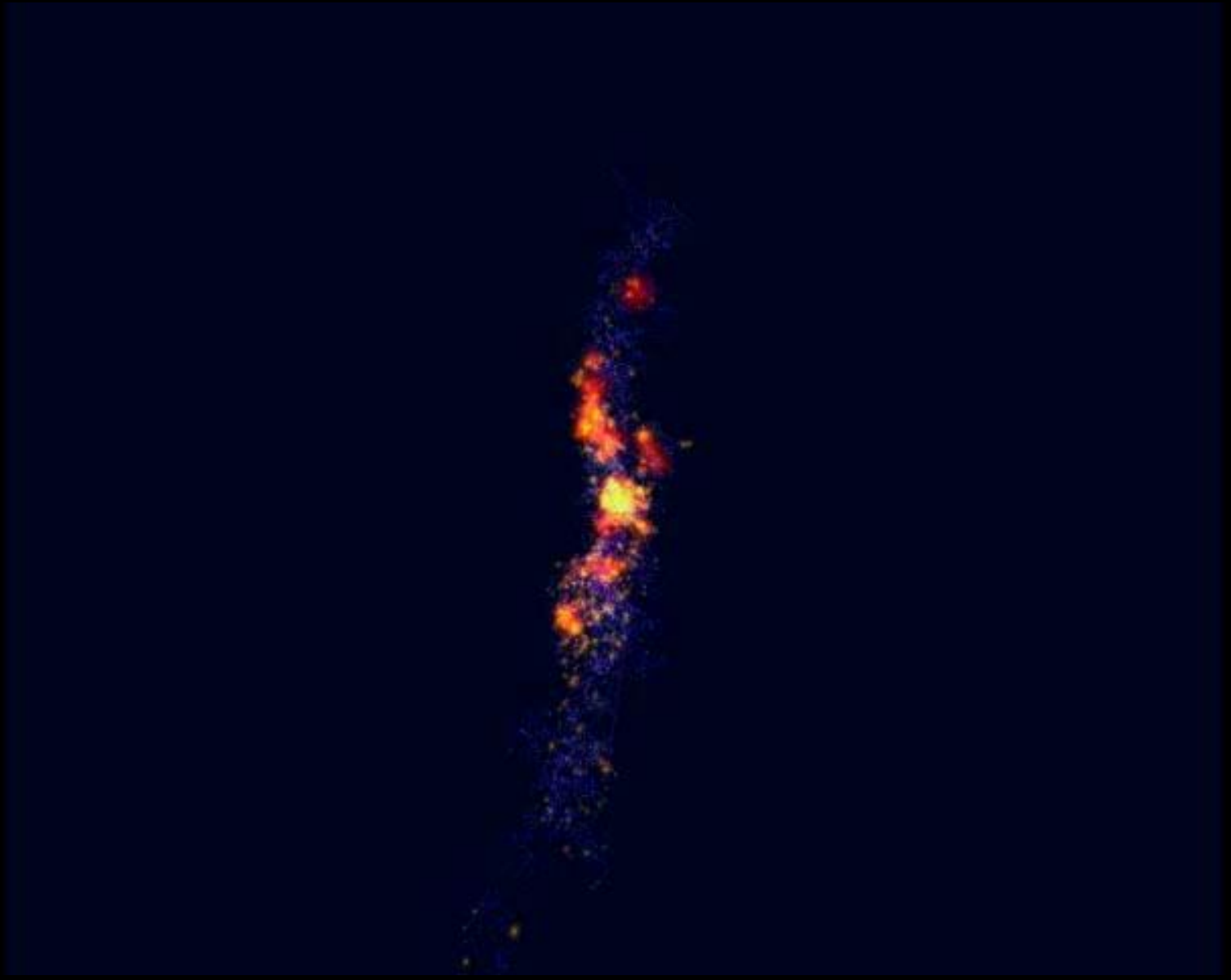


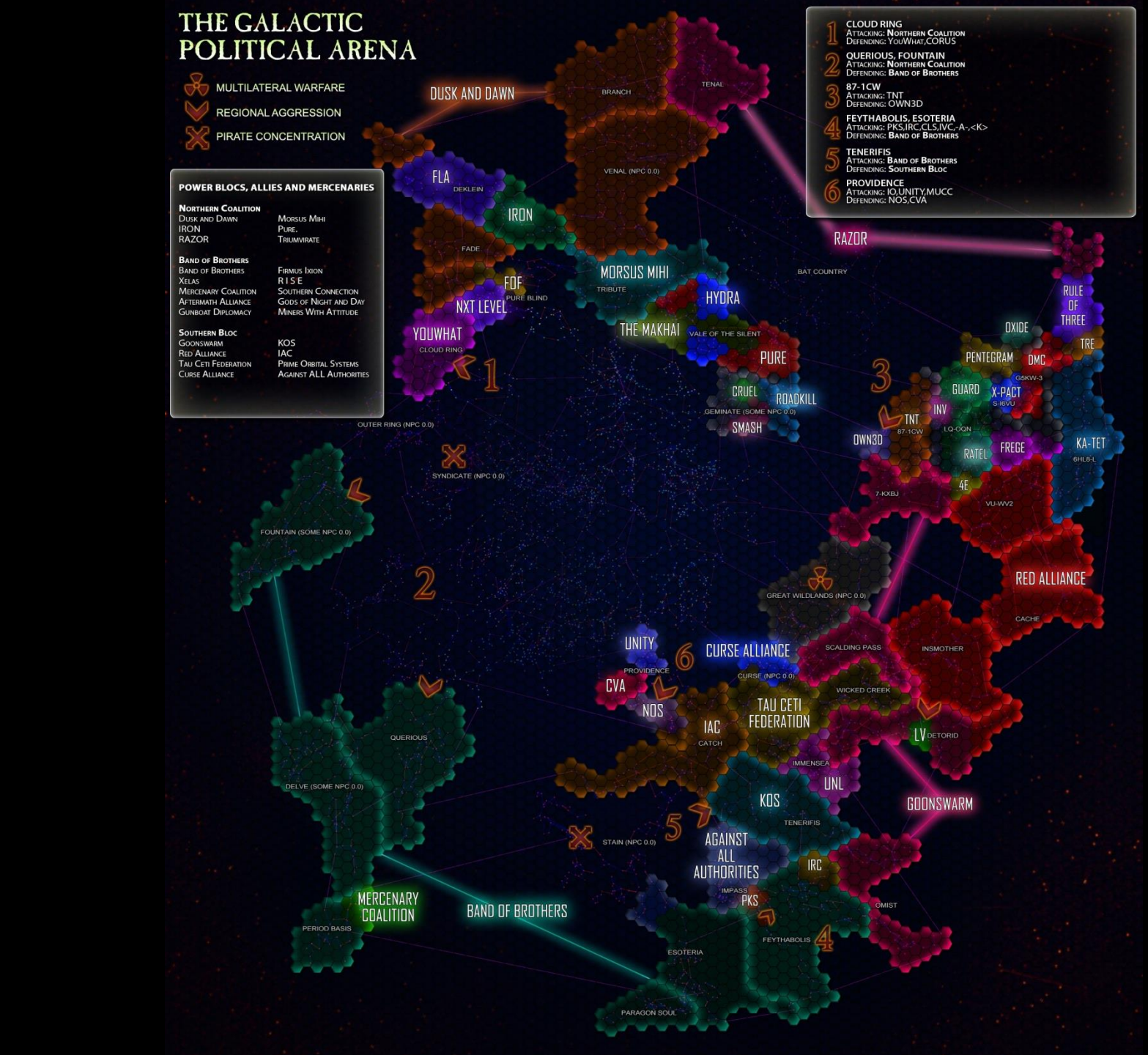
FIG. 1. TEM image of typical gold colloid aggregate. This cluster contains 4739 gold particles.




SMASH
(NPC 0.0)
UNITY


SMASH
OWN3D
(NPC 0.0)
7-KO
GREAT WILDLANDS (NPC 0.0)
UNITY
6
CURSE ALLIANCE
SCALDING PASS
PROVIDENCE
CURSE (NPC 0.0)
CVA
WICKED CREEK
NOS
TAU CETI
FEDERATION
IAC


SMASH
OWN3D
87-1CW
LQ-OQN
RATEL
FREGE
KA-T
6HL8-L
(NPC 0.0)
7-KXBJ
4E
VU-WV2
RED ALLIANCE
GREAT WILDLANDS (NPC 0.0)
CACHE
UNITY
6
CURSE ALLIANCE
SCALDING PASS
INSMOTHER
PROVIDENCE
CURSE (NPC 0.0)
CVA
NOS
WICKED CREEK
TAU CETI
FEDERATION
IAC
CATCH
LV
DETORID
IMMENSEA
UNL
KOS
GOONSWARM
TENERIFIS
5
STAIN (NPC 0.0)
AGAINST
ALL
AUTHORITIES
IRC

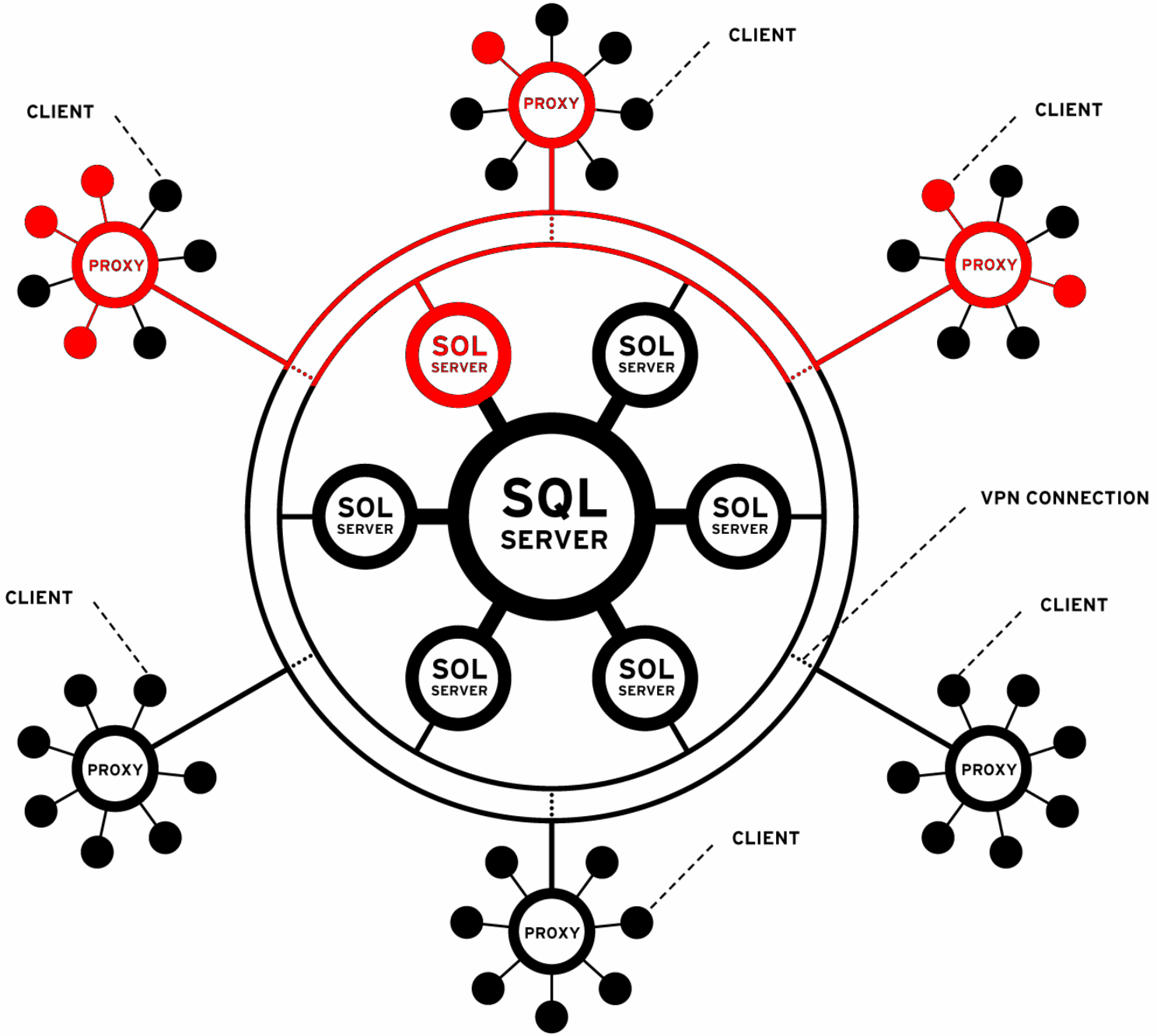

THE GALACTIC POLITICAL ARENA
MULTILATERAL WARFARE
REGIONAL AGGRESSION
PIRATE CONCENTRATION
POWER BLOCS, ALLIES AND MERCENARIES
Northern Coalition
Dusk and Dawn
IRON
RAZOR
Morsus Mihi
Pure.
Triumvirate
Band of Brothers
Band of Brothers
Xelas
Mercenary Coalition
Aftermath Alliance
Gunboat Diplomacy
Firmus Ixion
R I S E
Southern Connection
Gods of Night and Day
Miners With Attitude
Southern Bloc
Goonswarm
Red Alliance
Tau Ceti Federation
Curse Alliance
KOS
IAC
Prime Orbital Systems
Against ALL Authorities
1 CLOUD RING
Attacking: Northern Coalition
Defending: YouWhat,CORUS
2 QUERIOUS, FOUNTAIN
Attacking: Northern Coalition
Defending: Band of Brothers
3 87-1CW
Attacking: TNT
Defending: OWN3D
4 FEYTHABOLIS, ESOTERIA
Attacking: PKS,IRC,CLS,IVC,-A-,<K>
Defending: Band of Brothers
5 TENERIFIS
Attacking: Band of Brothers
Defending: Southern Bloc
6 PROVIDENCE
Attacking: IO,UNITY,MUCC
Defending: NOS,CVA
DUSK AND DAWN
BRANCH
TENAL
VENAL (NPC 0.0)
FLA
DEKLEIN
IRON
FADE
FOF
PURE BLIND
NXT LEVEL
YOUWHAT
CLOUD RING
MORSUS MIHI
TRIBUTE
HYDRA
THE MAKHAI
VALE OF THE SILENT
PURE
CRUEL
ROADKILL
GEMINATE (SOME NPC 0.0)
SMASH
OUTER RING (NPC 0.0)
SYNDICATE (NPC 0.0)
FOUNTAIN (SOME NPC 0.0)
QUERIOUS
DELVE (SOME NPC 0.0)
RAZOR
BAT COUNTRY
RULE OF THREE
OXIDE
TRE
PENTEGRAM
DMC
G5KW-3
GUARD
X-PACT
S-I6VU
INV
TNT
87-1CW
LQ-OQN
OWN3D
RATEL
FREGE
KA-TET
6HL8-L
4E
7-KXBJ
VU-WV2
RED ALLIANCE
CACHE
GREAT WILDLANDS (NPC 0.0)
UNITY
PROVIDENCE
CVA
NOS
CURSE ALLIANCE
CURSE (NPC 0.0)
SCALDING PASS
INSMOTHER
WICKED CREEK
IAC
CATCH
TAU CETI FEDERATION
LV
DETORID
IMMENSEA
UNL
KOS
TENERIFIS
GOONSWARM
STAIN (NPC 0.0)
AGAINST ALL AUTHORITIES
IMPASS
PKS
IRC
OMIST
FEYTHABOLIS
ESOTERIA
MERCENARY COALITION
PERIOD BASIS
BAND OF BROTHERS
PARAGON SOUL

Scaling well with increased hardware

- A unique combination of software and hardware layers
- Tiered architecture allows software to scale well with increased hardware
- Relies heavily on thread pooling and I/O completions, underneath micro-threaded Stackless Python logic

EVE runs on a supercomputer

Active Servers	~ 195
Weight	About 2.5 metric tonnes of equipment
CPUS	> 420 CPU cores > 1 THZ
RAM	> 7.5 Terabytes
Estimated FLOPS	> 7.5 Tera
Bandwidth	~ 400 GHZ


CLIENT
PROXY
CLIENT
PROXY
CLIENT
PROXY
SOL SERVER
SOL SERVER
SOL SERVER
SQL SERVER
SOL SERVER
VPN CONNECTION
CLIENT
PROXY
SOL SERVER
SOL SERVER
CLIENT
PROXY
CLIENT
PROXY