
GC
Est. 2011
Grand Cru
Kallio, Helsinki

Who the ?

- Roope Kangas - Lead Server Developer @ Grand Cru
- Company founded in April 2011
- Our first game was Supernauts in 2014
- Four new projects after that - working on two new titles now

What I am going to talk about

- Before Elixir
- How we build our projects
- Learnings and our experience

Supernauts

Supernauts

Supernauts

```
Thread 2 (Thread 0x7ff8772ff700 (LWP 2406)):
#0  0x00007ff960255ab1 in sem_timedwait () from /lib64/libpthread.so.0
#1  0x000000000061fb7b in mono_sem_timedwait (sem=0x965fc8 <async_tp+40>, ti
#2  0x0000000000585632 in async_invoke_thread (data=0x0) at threadpool.c:156
#3  0x000000000580de3 in start_wrapper_internal (data=0x7ff8c07d9000) at th
#4  start_wrapper (data=0x7ff8c07d9000) at threads.c:653
#5  0x0000000000614673 in thread_start_routine (args=0x7ff8cc0dd958) at wthre
#6  0x0000000000624340 in inner_start_thread (arg=0x7ff804de5d60) at mono-th
#7  0x00007ff96024ff18 in start_thread () from /lib64/libpthread.so.0
#8  0x00007ff95ff85e0d in clone () from /lib64/libc.so.6

Thread 1 (Thread 0x7ff960d7f780 (LWP 12092)):
#0  0x00007ff95fed72e2 in sigsuspend () from /lib64/libc.so.6
#1  0x00000000005c4714 in suspend_thread (info=0x1dbd040, context=0x7fff8850
#2  0x00000000005c486f in suspend_thread (context=0x7fff8850fb00, info=<optim
#3  suspend_handler (sig=<optimized out>, siginfo=<optimized out>, context=0x
#4  <signal handler called>
#5  0x00007ff95ff7ca3d in poll () from /lib64/libc.so.6
#6  0x00007ff937dd4bf3 in wait_for_any (shutting_down=<optimized out>, timeou
out>, signals=<optimized out>) at signal.c:355
#7  Mono_Unix_UnixSignal_WaitAny (_signals=0x7ff957fe7dc8, count=3, timeout=5
#8  0x00000000409aa723 in ?? ()
#9  0x0000000001e54630 in ?? ()
#10 0x00007ff960d7f700 in ?? ()
#11 0x0000000000000003 in ?? ()
#12 0x0000000041e3af76 in ?? ()
#13 0x00007ff957fe7dd8 in ?? ()
#14 0x00007ff957a12d78 in ?? ()
#15 0x00007fff88510310 in ?? ()
#16 0x0000000000000003 in ?? ()
#17 0x00007ff957fe7da8 in ?? ()
#18 0x00007ff957a127f0 in ?? ()
#19 0x0000000001e83ef0 in ?? ()
#20 0x0000000000000001 in ?? ()
#21 0x00007ff957fe7dc8 in ?? ()
#22 0x00007ff957fe7dc8 in ?? ()
#23 0x00007fff88510800 in ?? ()
#24 0x00000000409aa544 in ?? ()
#25 0x0000000001deea10 in ?? ()
#26 0x00007fff88510800 in ?? ()
#27 0x0000000000000000 in ?? ()

=================================================================
Got a SIGSEGV while executing native code. This usually indicates
a fatal error in the mono runtime or one of the native libraries
used by your application.
=================================================================
```

```
Heap shot summary

Heap shot 0 at 181.646 secs: size: 163489048, object count: 100534, class count: 2575, roots: 30733
     Bytes      Count  Average Class name
 152974568         78  1961212 System.Byte[]
         7 root references (4 pinning)
        24 references from: System.Byte[][]
         7 references from: System.IO.FileStream
         5 references from: MongoDB.Bson.IO.BsonTrieNode<System.Int32>
         4 references from: System.IO.StreamWriter
         4 references from: System.Security.Cryptography.MD5CryptoServiceProvider
         3 references from: System.IO.CStreamWriter
         3 references from: Mono.Globalization.Unicode.SimpleCollator
         2 references from: GrandCru.FastZlib.ZStream
         2 references from: System.MonoCQItem
         2 references from: log4net.Appender.TurfServerS3Appender
         1 references from: LookupService
         1 references from: System.IO.MemoryStream
         1 references from: System.TermInfoReader
         1 references from: GameConfig
         1 references from: System.IO.CStreamReader
         1 references from: MongoDB.Bson.IO.BsonTrieNode<System.Boolean>
         1 references from: System.Reflection.Emit.ModuleBuilder
         1 references from: MongoDB.Bson.IO.BsonChunk
         1 references from: GrandCru.FastZlib.Inf8locks
         1 references from: StaticPool.poolentry[]
   3522368      28080       125 System.String
      5091 root references (2 pinning)
     24861 references from: System.String[]
     14058 references from: GeneralTaskConfig
      8015 references from: GrandCru.Util.StatAverages.Counter[]
      6702 references from: Utils.StaticEntry5
      5174 references from: System.Object[]
      4470 references from: Cache.Entry<KTuple<System.String,System.String>,Utils.StaticEntry5>
      3476 references from: Utils.StaticEntry6
      2613 references from: Cache.Entry<KTuple<System.String,System.String,System.String>,Utils.StaticEntry6>
      2510 references from: STuple<System.String,System.String,System.Type>[]
      1000 references from: System.Reflection.MonoField
       762 references from: ObjectSerializationHelper.SerializableFieldInfo
       624 references from: ChatEntry
       581 references from: MissionProgressionData
       435 references from: Substance
       400 references from: MongoDB.Bson.Serialization.BsonMemberMap
       390 references from: Utils.StaticEntry1
       384 references from: TurfObjectConfig
       361 references from: Cache.Entry<System.String,Utils.StaticEntry12>
       360 references from: Utils.StaticEntry12
       357 references from: System.Reflection.MonoParameterInfo
       278 references from: System.Collections.Hashtable.Slot[]
       252 references from: System.Configuration.SectionInfo
       200 references from: ShopConfigEntry
       196 references from: Cache.Entry<KTuple<System.String,System.Int32>,Utils.StaticEntry1>
       186 references from: System.MonoTypeInfo
       167 references from: System.Text.StringBuilder
       146 references from: System.Collections.Specialized.NameObjectCollectionBase._Item
       129 references from: SquareUpgradeConfig
```

Size: 117 KB Earlier Later Now

GC Est. 2011 Grand Cru Kallio, Helsinki

Supernauts

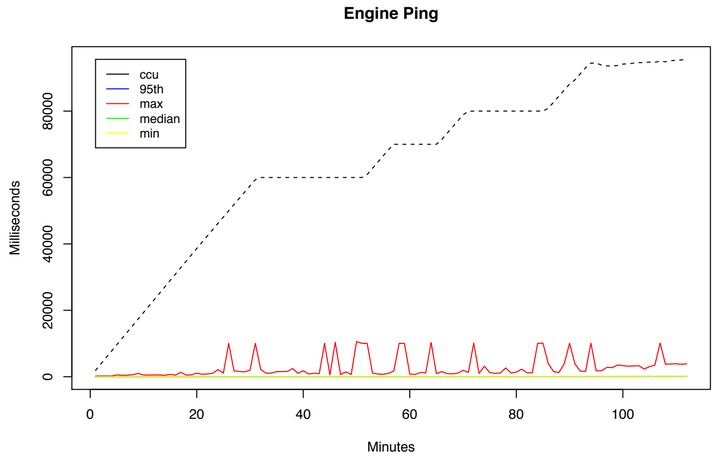
```
Thread 2 (Thread 0x7ff8772ff700 (LWP 2406)):
#0  0x00007ff960255ab1 in sem_timedwait () from /lib64/libpthread.so.0
#1  0x000000000061fb7b in mono_sem_timedwait (sem=0x965fc8 <async_tp+40>, ti
#2  0x0000000000585632 in async_invoke_thread (data=0x0) at threadpool.c:156
#3  0x00000000000580de3 in start_wrapper_internal (data=0x7ff8c07d9000) at th
#4  start_wrapper (data=0x7ff8c07d9000) at threads.c:653
#5  0x0000000000614673 in thread_start_routine (args=0x7ff8cc0dd958) at wthre
#6  0x0000000000624340 in inner_start_thread (arg=0x7ff804de5d60) at mono-thr
#7  0x00007ff96024ff18 in start_thread () from /lib64/libpthread.so.0
#8  0x00007ff95ff85e0d in clone () from /lib64/libc.so.6

Thread 1 (Thread 0x7ff960d7f780 (LWP 12092)):
#0  0x00007ff95fed72e2 in sigsuspend () from /lib64/libc.so.6
#1  0x00000000005c4714 in suspend_thread (info=0x1dbd040, context=0x7fff8850
#2  0x00000000005c486f in suspend_thread (context=0x7fff8850fb00, info=<optim
#3  suspend_handler (sig=<optimized out>, siginfo=<optimized out>, context=0x
#4  <signal handler called>
#5  0x00007ff95ff7ca3d in poll () from /lib64/libc.so.6
#6  0x00007ff937dd4bf3 in wait_for_any (shutting_down=<optimized out>, timeou
out>, signals=<optimized out>) at signal.c:355
#7  Mono_Unix_UnixSignal_WaitAny (_signals=0x7ff957fe7dc8, count=3, timeout=5
#8  0x00000000409aa723 in ?? ()
#9  0x0000000001e54630 in ?? ()
#10 0x00007ff960d7f700 in ?? ()
#11 0x0000000000000003 in ?? ()
#12 0x0000000041e3af76 in ?? ()
#13 0x00007ff957fe7dd8 in ?? ()
#14 0x00007ff957a12d78 in ?? ()
#15 0x00007fff88510310 in ?? ()
#16 0x0000000000000003 in ?? ()
#17 0x00007ff957fe7da8 in ?? ()
#18 0x00007ff957a127f0 in ?? ()
#19 0x0000000001e83ef0 in ?? ()
#20 0x0000000000000001 in ?? ()
#21 0x00007ff957fe7dc8 in ?? ()
#22 0x00007ff957fe7dc8 in ?? ()
#23 0x00007fff88510800 in ?? ()
#24 0x00000000409aa544 in ?? ()
#25 0x0000000001deea10 in ?? ()
#26 0x00007fff88510800 in ?? ()
#27 0x0000000000000000 in ?? ()

=================================================================
Got a SIGSEGV while executing native code. This usually indicates
a fatal error in the mono runtime or one of the native libraries
used by your application.
=================================================================
```

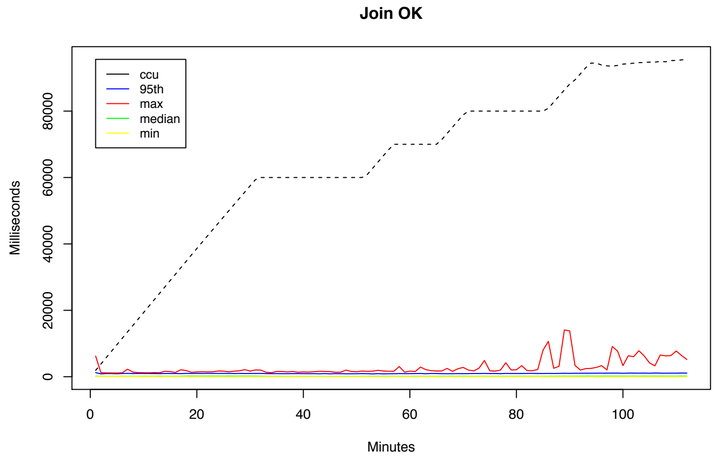
```
Heap shot summary

Heap shot 0 at 181.646 s
      Bytes       Count  A
  152974568          78  1
          7 root reference
         24 references fr
          7 references fr
          5 references fr
          4 references fr
          4 references fr
          3 references fr
          3 references fr
          2 references fr
          2 references fr
          2 references fr
          1 references fr
          1 references fr
          1 references fr
          1 references fr
          1 references fr
          1 references fr
          1 references fr
          1 references fr
          1 references fr
          1 references fr
    3522368      28080
       5091 root refere
      24861 reference
      14058 reference
       8015 reference
       6702 reference
       5174 reference
       4470 reference
       3476 reference
       2613 references from: Cache.Entry<KTuple<System.String,System.String,System.String>,Utils.StaticEntry6>
       2510 references from: STuple<System.String,System.String,System.Type>[]
       1000 references from: System.Reflection.MonoField
        762 references from: ObjectSerializationHelper.SerializableFieldInfo
        624 references from: ChatEntry
        581 references from: MissionProgressionData
        435 references from: Substance
        400 references from: MongoDB.Bson.Serialization.BsonMemberMap
        390 references from: Utils.StaticEntry1
        384 references from: TurfObjectConfig
        361 references from: Cache.Entry<System.String,Utils.StaticEntry12>
        360 references from: Utils.StaticEntry12
        357 references from: System.Reflection.MonoParameterInfo
        278 references from: System.Collections.Hashtable.Slot[]
        252 references from: System.Configuration.SectionInfo
        200 references from: ShopConfigEntry
        196 references from: Cache.Entry<KTuple<System.String,System.Int32>,Utils.StaticEntry1>
        186 references from: System.MonoTypeInfo
        167 references from: System.Text.StringBuilder
        146 references from: System.Collections.Specialized.NameObjectCollectionBase._Item
        129 references from: SquareUpgradeConfig
```

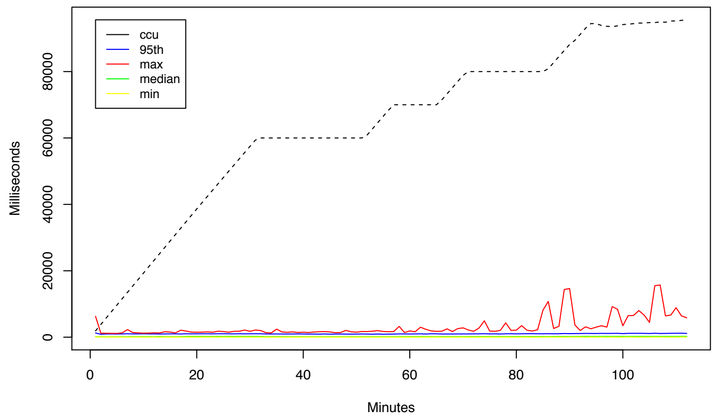
Size: 117 KB Earlier Later Now

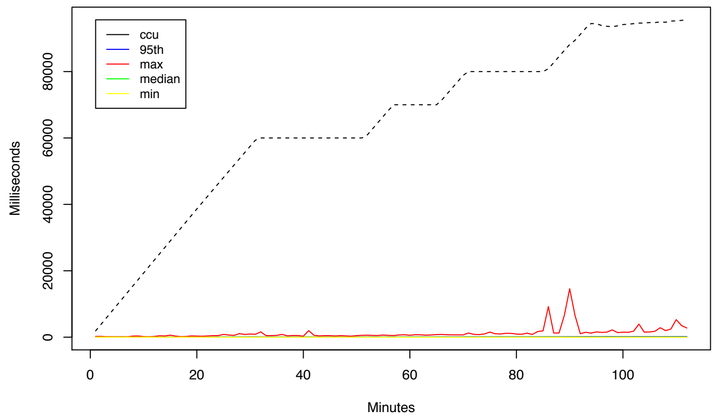
GC Est. 2011 Grand Cru Kallio, Helsinki

Used Erlang to load-test Supernauts









Learnings from Supernauts

Learnings from Supernauts

- Required a lot of "plumbing" to manage concurrency

Learnings from Supernauts

- Required a lot of "plumbing" to manage concurrency

Learnings from Supernauts

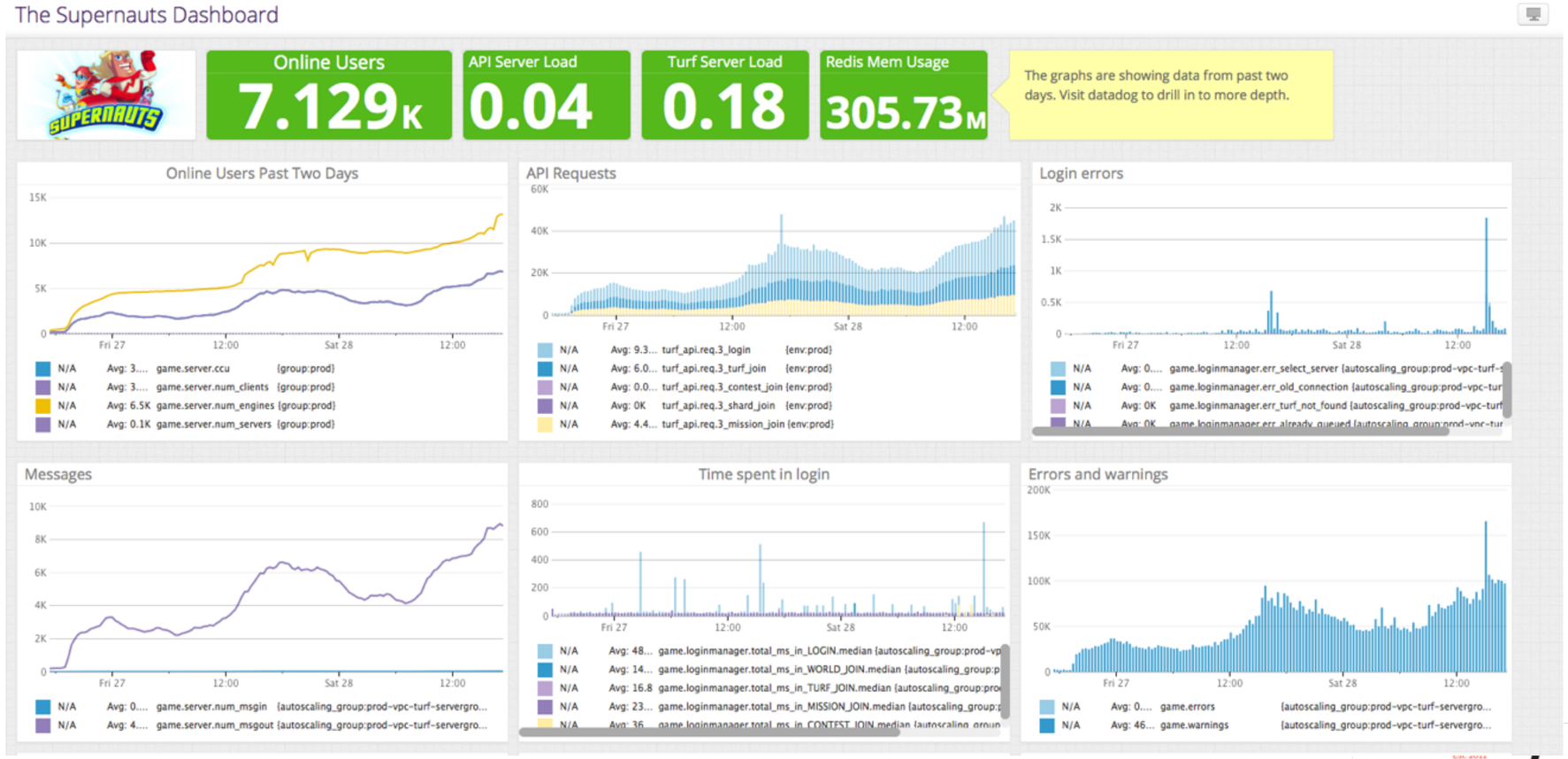
- Required a lot of "plumbing" to manage concurrency
- Chosen tech did not feel like a natural fit for the backend

Learnings from Supernauts

- Required a lot of "plumbing" to manage concurrency
- Chosen tech did not feel like a natural fit for the backend
  - Ended up doing lightweight processes with mailboxes by ourselves... :D

Supernauts

Supernauts



Our first sips

1) http://media.webcollage.net/rwvfp/wc/cp/19805765/module/tommeetippeeus/_cp/products/1420707514524/tab-6335b7db-4e6a-4b4a-89a4-c9babc7d2d19/dfbf5268-54fe-4457-bc91-8056bd7bde2b.jpg.w240.jpg

2) http://cdn.babygearlab.com/photos/0/58/306805_22066_XL.jpg

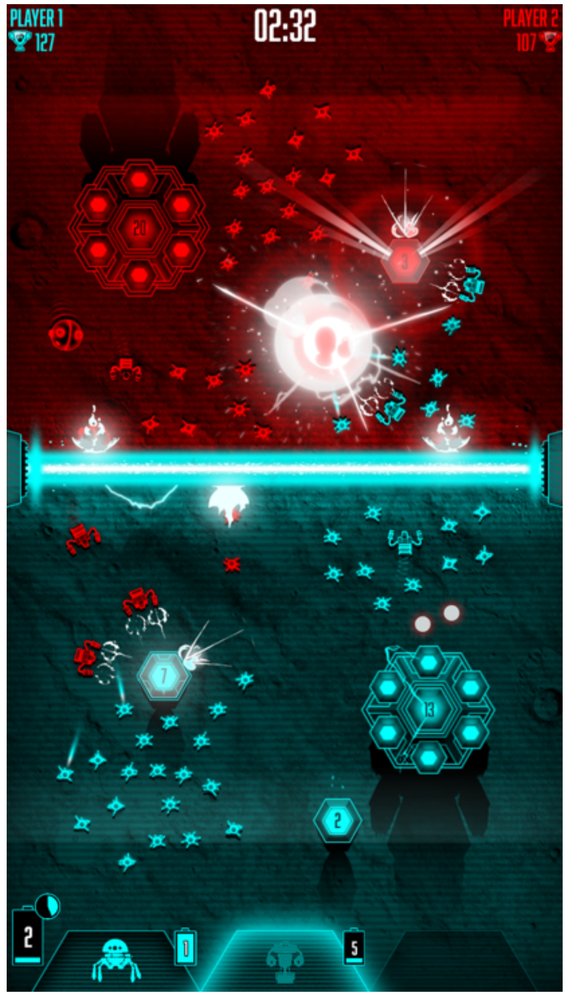
Warpfield 2014 <-> 2015

Warpfield - overview

- Player vs Player
- Crews and Tiered Leagues
- Free-to-play game
- Always connected

Grand Cru
Est. 2011
Kallio, Helsinki

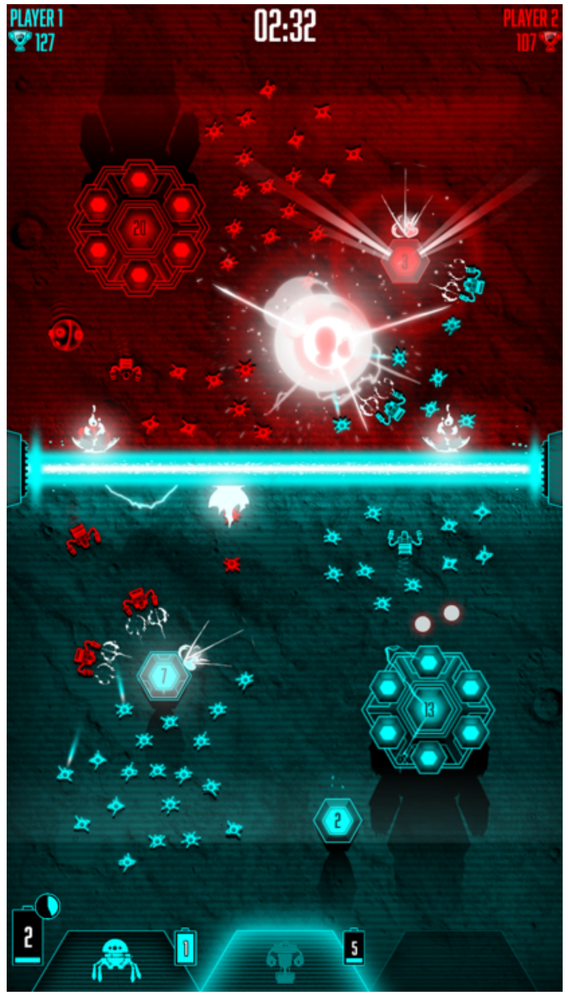
Warpfield - overview



- Player vs Player
- Crews and Tiered Leagues
- Free-to-play game
- Always connected

Grand Cru
Kallio, Helsinki

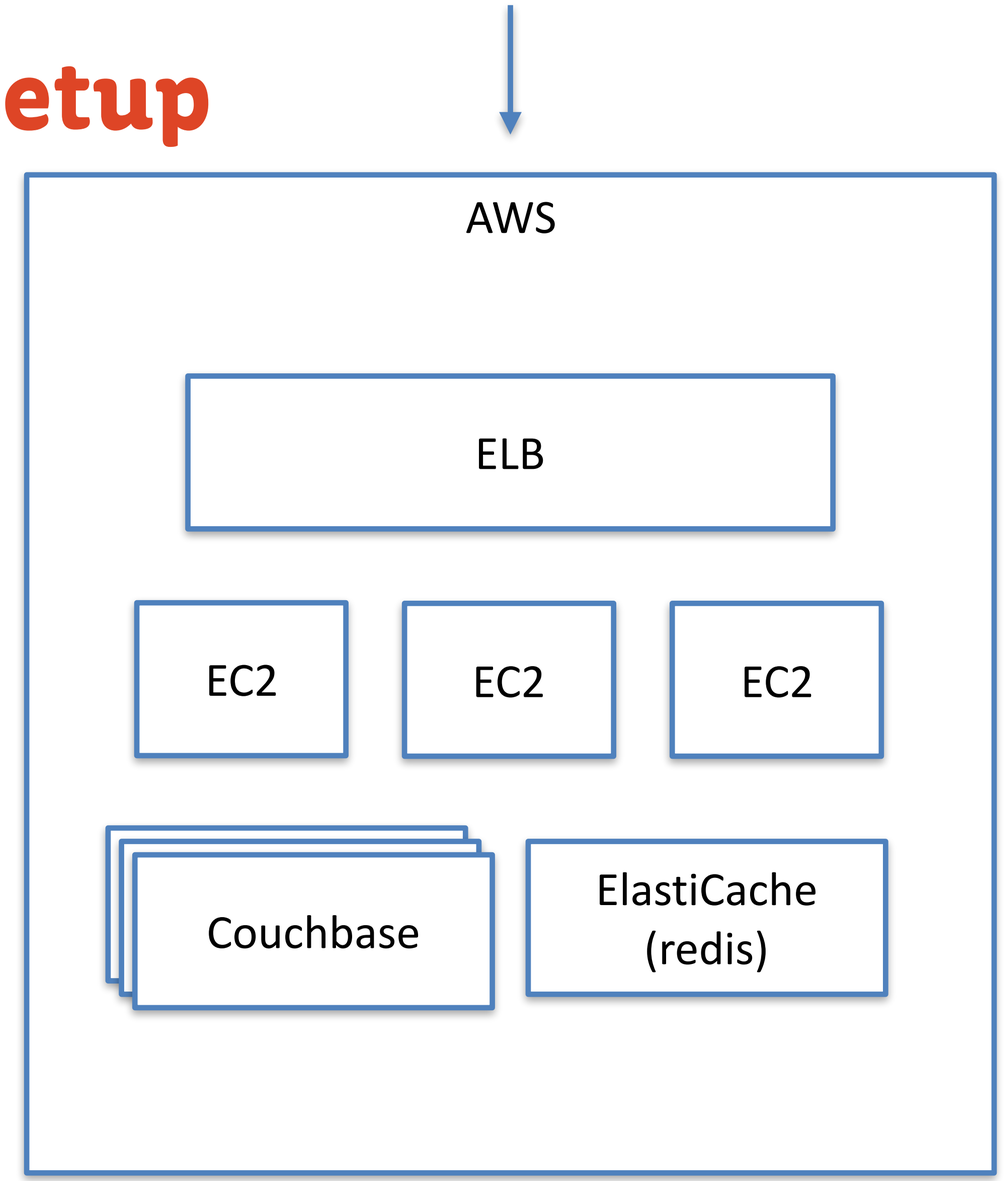
Warpfield - overview



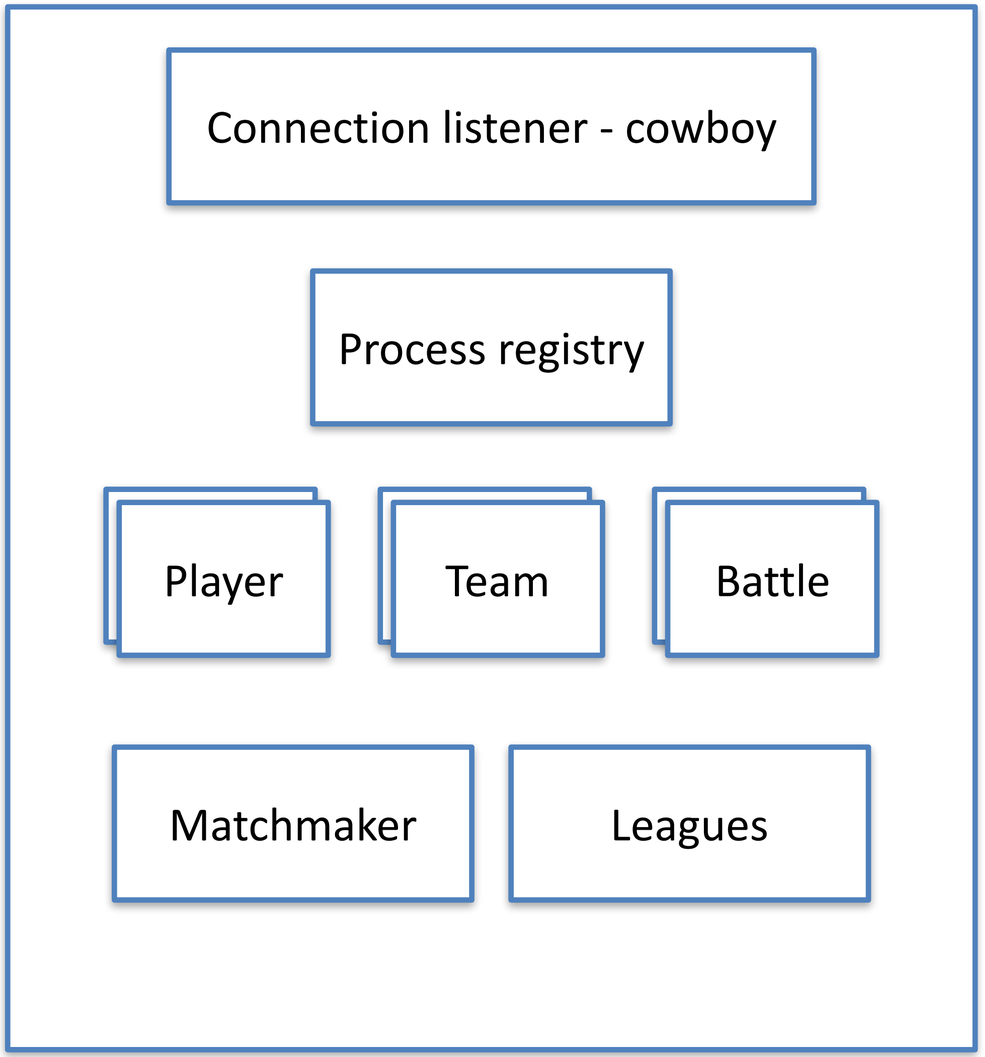
- Player vs Player
- Crews and Tiered Leagues
- Free-to-play game
- Always connected

Grand Cru
Kallio, Helsinki

Server setup



Inside the server



Finding the game process

Finding the game process

- GenServer supports custom registries

Finding the game process

- GenServer supports custom registries

Finding the game process

- GenServer supports custom registries

```
GenServer.start_link(__MODULE__, {:via, Relocker.Registry, name})
```

Finding the game process

- GenServer supports custom registries

```
GenServer.start_link(__MODULE__, {:via, Relocker.Registry, name})
```

Finding the game process

- GenServer supports custom registries

```
GenServer.start_link(__MODULE__, {:via, Relocker.Registry, name})


# Node 1
```

Finding the game process

- GenServer supports custom registries

```
GenServer.start_link(__MODULE__, {:via, Relocker.Registry, name})


# Node 1
{:ok, pid} = Game.start_link([], name: "player_1")
```

Finding the game process

- GenServer supports custom registries

```
GenServer.start_link(__MODULE__, {:via, Relocker.Registry, name})


# Node 1
{:ok, pid} = Game.start_link([], name: "player_1")
```

Finding the game process

- GenServer supports custom registries

```
GenServer.start_link(__MODULE__, {:via, Relocker.Registry, name})


# Node 1
{:ok, pid} = Game.start_link([], name: "player_1")
```

Finding the game process

- GenServer supports custom registries

```
GenServer.start_link(__MODULE__, {:via, Relocker.Registry, name})


# Node 1
{:ok, pid} = Game.start_link([], name: "player_1")
```

Finding the game process

- GenServer supports custom registries

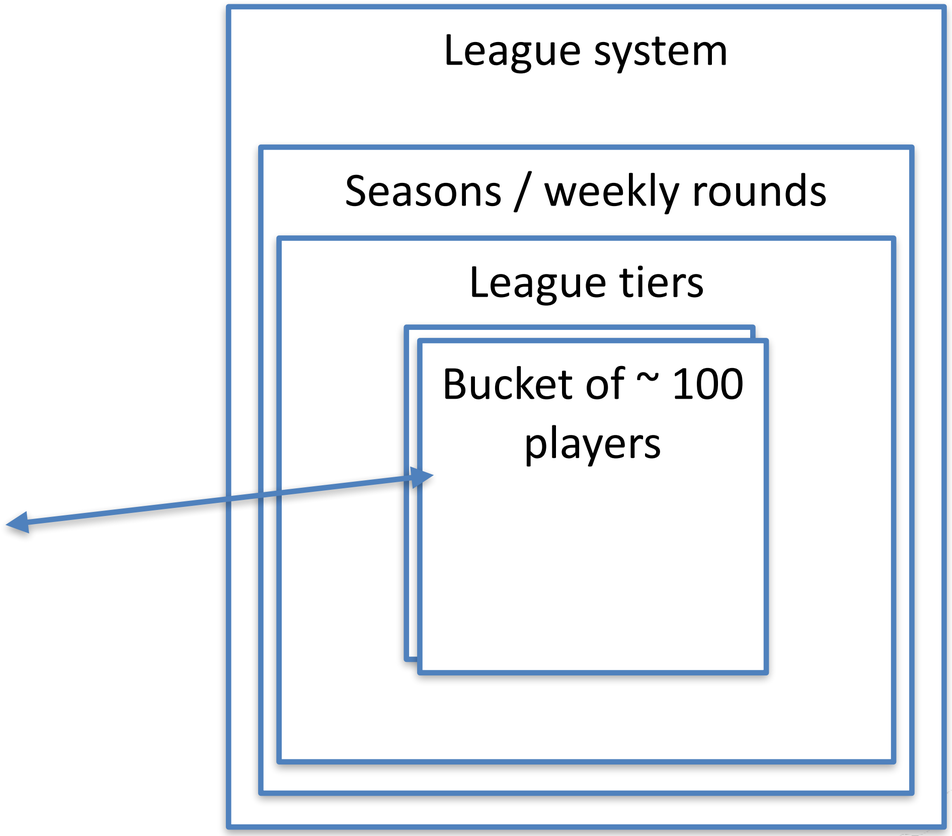
```
GenServer.start_link(__MODULE__, {:via, Relocker.Registry, name})


# Node 1
{:ok, pid} = Game.start_link([], name: "player_1")



# Node 2
```

Finding the game process

- GenServer supports custom registries

```
GenServer.start_link(__MODULE__, {:via, Relocker.Registry, name})


# Node 1
{:ok, pid} = Game.start_link([], name: "player_1")



# Node 2
GenServer.cast({:via, Relocker.Registry, "player_1"}, :stop)
```

League system



Implementing Leagues with Elixir

Implementing Leagues with Elixir

Implementing Leagues with Elixir

- Built first without side-effects, sketch out the logic and tests

Implementing Leagues with Elixir

- Built first without side-effects, sketch out the logic and tests

Implementing Leagues with Elixir

- Built first without side-effects, sketch out the logic and tests
- For side-effects used in-memory implementation first with an Agent

Implementing Leagues with Elixir

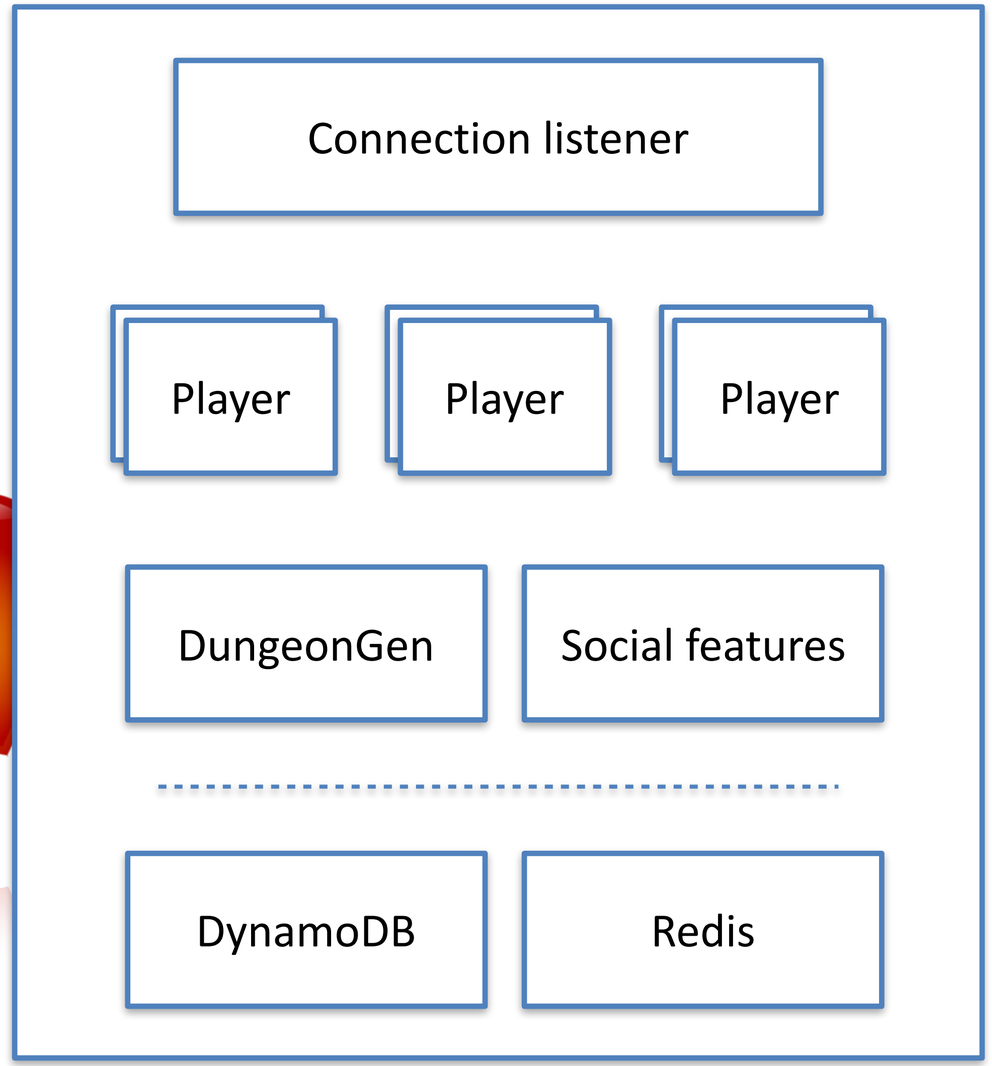
- Built first without side-effects, sketch out the logic and tests
- For side-effects used in-memory implementation first with an Agent

Implementing Leagues with Elixir

- Built first without side-effects, sketch out the logic and tests
- For side-effects used in-memory implementation first with an Agent
- Divide and conquer!

Hero Hack

Hero Hack Server



Hero Hack - client <-> server sync

Hero Hack - client <-> server sync

- Wire-protocol generated from Protobuf specification

Hero Hack - client <-> server sync

- Wire-protocol generated from Protobuf specification

Hero Hack - client <-> server sync

- Wire-protocol generated from Protobuf specification

Hero Hack - client <-> server sync

```
message C2S_Auth {
    optional string device_model = 1;
    optional string device_id = 2;
    optional string config_hash = 3;
    optional string device_name = 4;
    optional bool fake_recap_events = 5;
    optional string version = 6;
    optional int64 client_time = 7;
}
```

Hero Hack - client <-> server sync

```
var request = GameAPI.Instance.Auth(new C2S_Auth{...});
yield return request.Start(); // wait for reply
var response = request.Response;
if (response.Ok) {
  // \o/
} else {
  // :(
}
```

Hero Hack - client <-> server sync

```
# Websocket process
def handle_message(%C2S_Auth{} = msg, req, %{state: :auth} = state) do
 player_id = PlayerRegistry.device_to_player!(msg.device_id)
 {:ok, pid} = GameSupervisor.find_or_start_game(args)
 GameProcess.handle_message(pid, msg)
end
```

Live config updates in dev

Live config updates in dev

- https://github.com/GrandCru/GoogleSheets

Live config updates in dev

- https://github.com/GrandCru/GoogleSheets

Live config updates in dev

- https://github.com/GrandCru/GoogleSheets
  - Poll config updates from Google Sheets to the game server

Live config updates in dev

- https://github.com/GrandCru/GoogleSheets
  - Poll config updates from Google Sheets to the game server

Live config updates in dev

- https://github.com/GrandCru/GoogleSheets
  - Poll config updates from Google Sheets to the game server
  - Used in as part of our development process, production stuff separated

Elixir generated dungeons

Elixir generated dungeons

```
54  rule medium_rooms:
55                      wall_length     4..7
56
57  rule large_rooms:
58                      wall_length     5..15
59
60
61  # DINGY DUNGEONS ==========================================================
62
63  area 1:
64                      medium_rooms
65                      width           20..25
66                      height          20..25
67                      proplevel       1
68                      propamount      2
69                      clutterlevel    1
70                      hpower          4
71                      door_chance     30
72                      locked_door_chance  15
73                      ammolevel       1
74
75                      dungeon 1:
76                                      stages          2
77                                      mlevel          0
78                                      mpower          6..8
79                                      light           100
80                                      ammoprob        0%
81                                      locked_door_chance  0
82                                      chestlevel      1
```

Elixir generated dungeons

```elixir
def generate_stage(conf, %GameConfig{} = gameconfig, rnd, opts \\ []) do
  # stage size
  width = number_or_random_range(rnd, conf.width)
  height = number_or_random_range(rnd, conf.height)

  # generate a random bsp and rooms from that
  bsp = BSPGen.generate_bsp(conf, rnd, {0, 0, width, height}, 6)
  rooms = generate_rooms(conf, rnd, gameconfig, bsp)

  # choose rooms which may contain traps and have the different floor type.
  # this is a subset of all rooms.
  rooms_with_traps = generate_rooms_with_traps(conf, rnd, gameconfig, rooms)

  # generate pathways to connect rooms
  pathways = generate_pathways(conf, rnd, rooms)
```

Elixir generated dungeons

Elixir generated dungeons

Load testing - setup

Load testing - setup

- Custom client application

Load testing - setup

- Custom client application
  - Simulates the game client, sends a fews messages - mostly idle

Load testing - setup

- Custom client application
  - Simulates the game client, sends a fews messages - mostly idle

Load testing - setup

- Custom client application
  - Simulates the game client, sends a fews messages - mostly idle

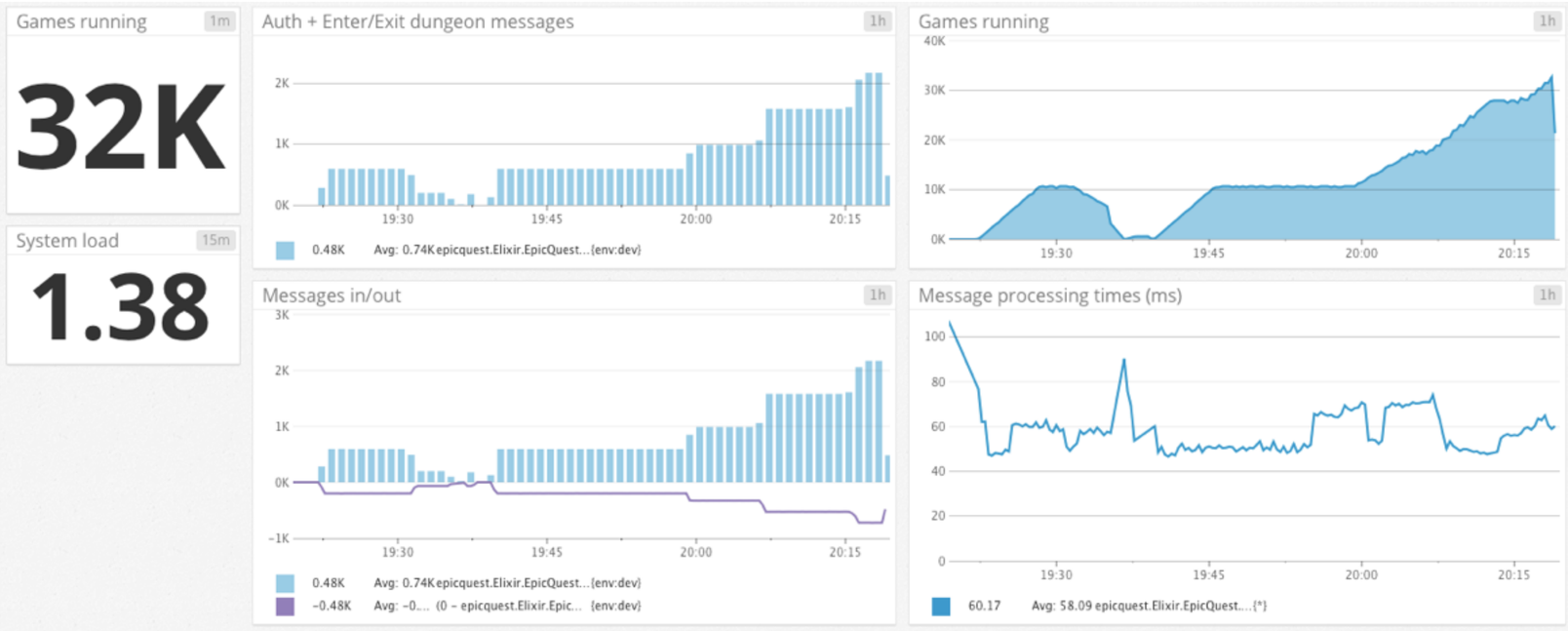
- Game servers **2** r3.xlarge 4 cores + 30GB RAM

Load testing - setup

- Custom client application
  - Simulates the game client, sends a fews messages - mostly idle
- Game servers **2** r3.xlarge 4 cores + 30GB RAM
- DynamoDB 200/200 read/write ops for our devices & player tables

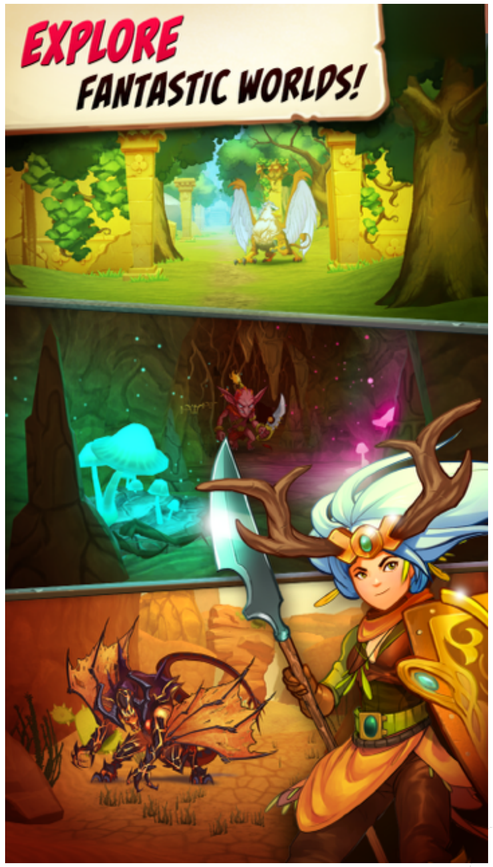
Load testing

Load testing



Battlejack

Battlejack



Grand Cru
Est. 2011
Kallio, Helsinki

From client driven to server drive

From client driven to server drive

- Client still predicts forward and assumes all operations succeed

From client driven to server drive

- Client still predicts forward and assumes all operations succeed

From client driven to server drive

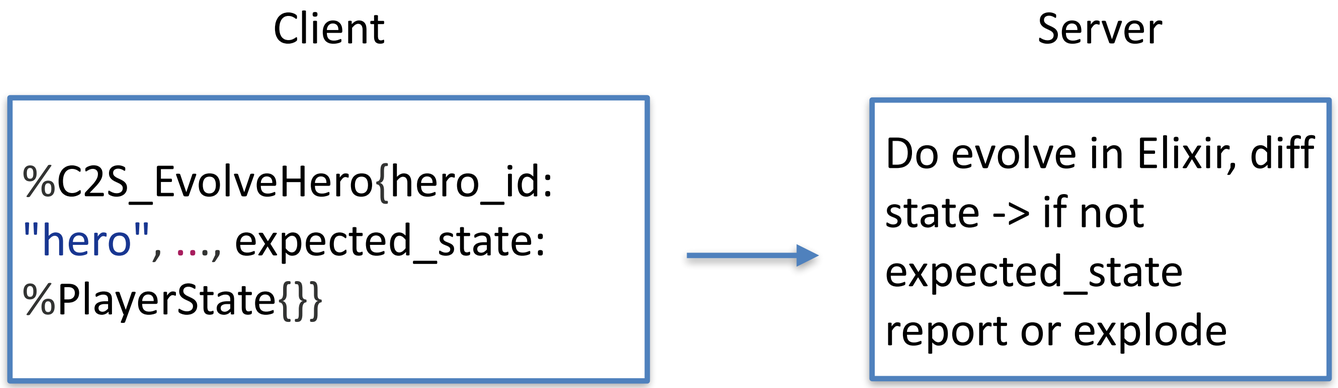
- Client still predicts forward and assumes all operations succeed

- But while we are still working on the server, the action and resulting player state is sent to server

From client driven to server drive

- Client still predicts forward and assumes all operations succeed

- But while we are still working on the server, the action and resulting player state is sent to server



Server setup & tools

Server setup & tools

- Everything on AWS

Server setup & tools

- Everything on AWS
  - ELB, EC2, DynamoDB, Redis

Server setup & tools

- Everything on AWS
  - ELB, EC2, DynamoDB, Redis

Server setup & tools

- Everything on AWS
  - ELB, EC2, DynamoDB, Redis

- Jenkins, builds releases

Server setup & tools

- Everything on AWS
  - ELB, EC2, DynamoDB, Redis

- Jenkins, builds releases

Server setup & tools

- Everything on AWS
  - ELB, EC2, DynamoDB, Redis
- Jenkins, builds releases
- Ansible for setup and deployment

Server setup & tools

- Everything on AWS
  - ELB, EC2, DynamoDB, Redis
- Jenkins, builds releases
- Ansible for setup and deployment

Server setup & tools

- Everything on AWS
  - ELB, EC2, DynamoDB, Redis
- Jenkins, builds releases
- Ansible for setup and deployment
- Monitoring tools:

Server setup & tools

- Everything on AWS
  - ELB, EC2, DynamoDB, Redis
- Jenkins, builds releases
- Ansible for setup and deployment
- Monitoring tools:
  - Datadog, Pingdom, AWS Cloudwatch

Elixir in our experience

Elixir in our experience

- Easy to get started

Elixir in our experience

- Easy to get started
  - Every coder in the company can change the server

Elixir in our experience

- Easy to get started
  - Every coder in the company can change the server

Elixir in our experience

- Easy to get started
  - Every coder in the company can change the server

- Easy to build reliable software

Elixir in our experience

- Easy to get started
  - Every coder in the company can change the server

- Easy to build reliable software

Elixir in our experience

- Easy to get started
  - Every coder in the company can change the server
- Easy to build reliable software
- Elixir promotes code reuse - at least we have been able to share code between projects


GC
Est. 2011
Thank You!