オンラインゲームサービスにおける MySQL活用事例

2002・2　コミュニティエンジン株式会社　　中嶋謙互

オンラインゲーム の特徴

オフラインゲームにくらべると…

- 身体能力を発揮して楽しむより、思考を楽しむ内容に向く（光速の限界）
- 自分との戦いよりもコミュニケーションの楽しさを求める内容に向く
- 静的な世界よりも動的な世界観に向く

ゲーム開発 vs. 一般の開発

ゲーム開発		一般の開発
生活に必要ではない	←→	生活の役に立つ
「それを作りたい」から始まる	←→	「それが必要」から始まる
すぐに飽きられる	←→	役立つ限り、使い続ける

★ゲームは何の問題も解決しない。

典型的に違う例：予算配分

ゲーム開発の場合

一般の開発の場合

A ： 面白くするための部分
B ： 見た目をよくするための部分
C ： 性能（機能）をよくするための部分
D ： 信頼性を高めるための部分

繰り返されてきた議論

「ゲーム用にこういう性能のシステムがほしい」

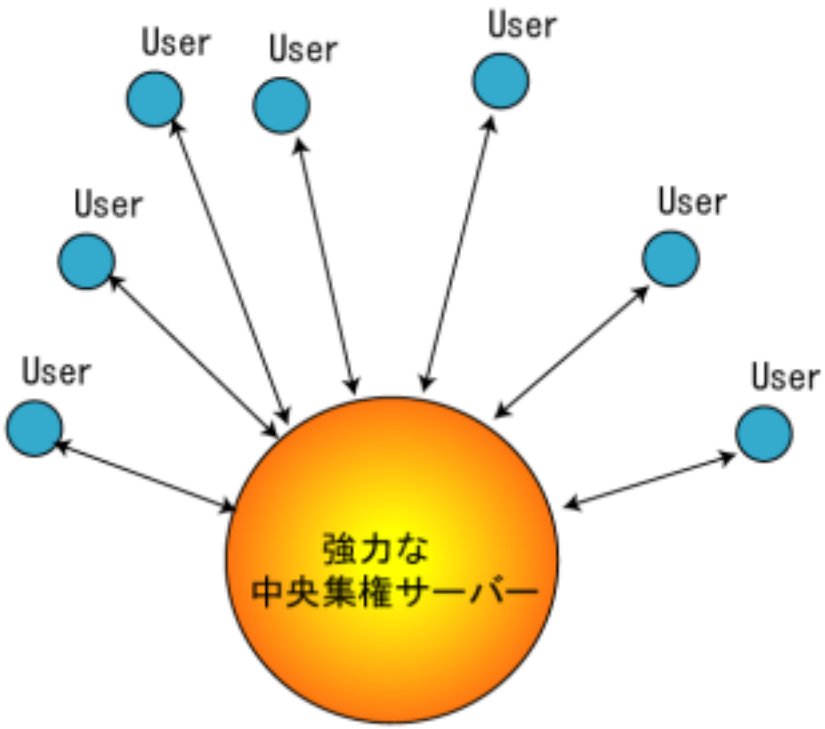
→ 一般のシステム開発会社に見積もりを依頼
→ N億円かかると言われる
→ 信頼性はいらないのでN百万円でやってくれと交渉する
→ それは無理だと言われる
→ 仕方なく自分でやることにする
→ オープンソースのツールを使いつつ自分で作る
→ できてしまいました

オンラインゲームの3類型

* Massive Multiplayer Online game
* Peer to Peer Network game
* Hybrid Network game

MMO型ゲーム

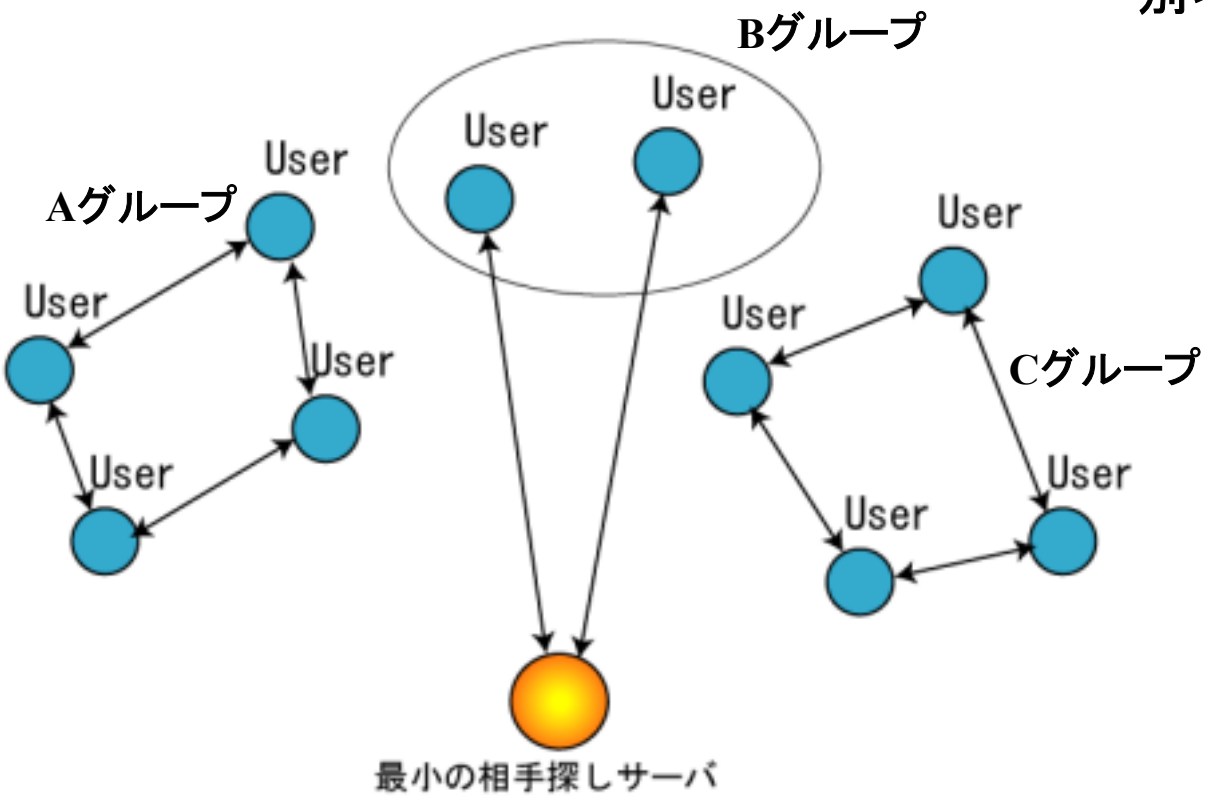
別名：OnlineGame



ユーザー数：数千～数万
オフラインで遊べない

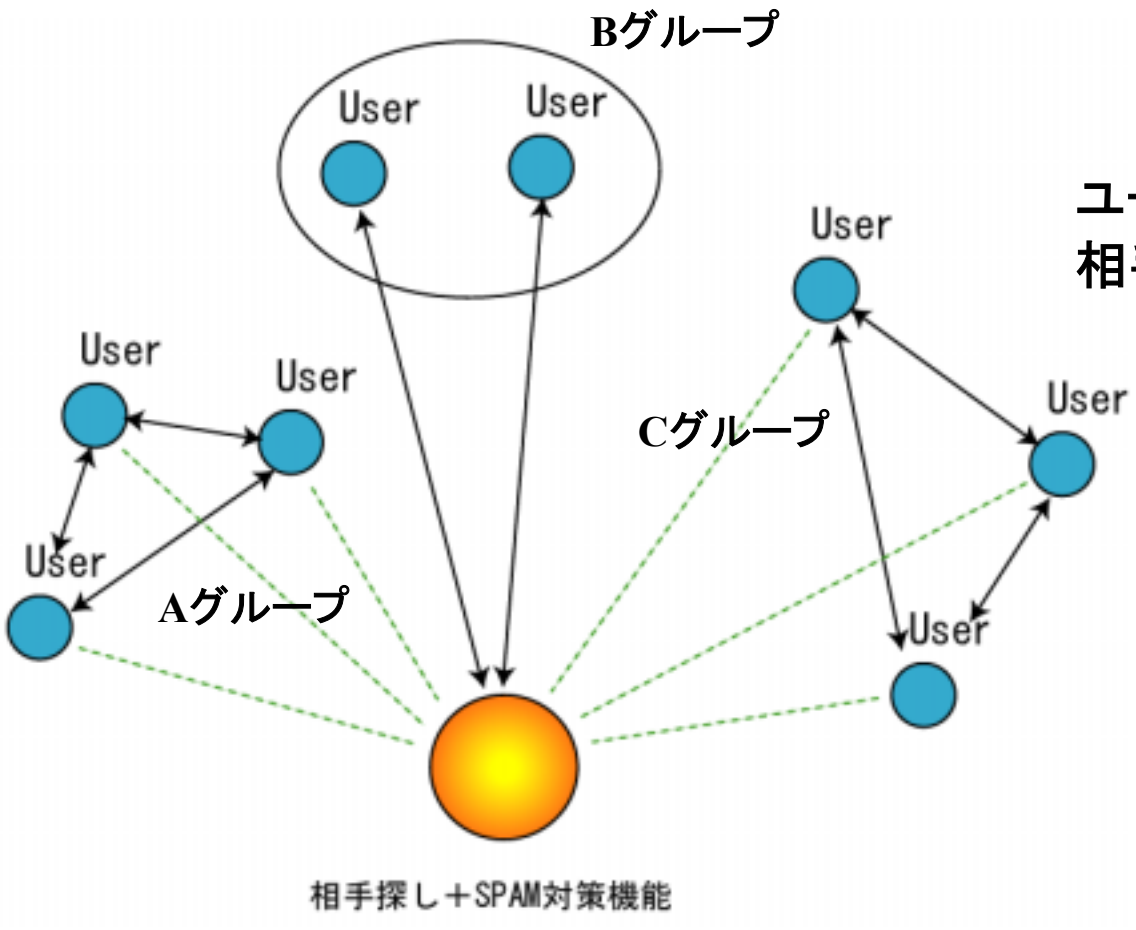
P2P型ゲーム

別名：NetworkGame



ユーザー数：不明
オフラインで遊べる

ハイブリッド型ゲーム



別名：HybridNetworkGame

ユーザー数：MMOとP2Pの中間
相手探し後も多少通信する

今後の動向

ゲームプレイの品質向上を求めるため、
ハイブリッドもしくはMMO型が主流になっていく。

→　ますますバックエンドの重要性が高まる

オープンソースソフトウェアとゲームシステム開発の相性

相性のよい点

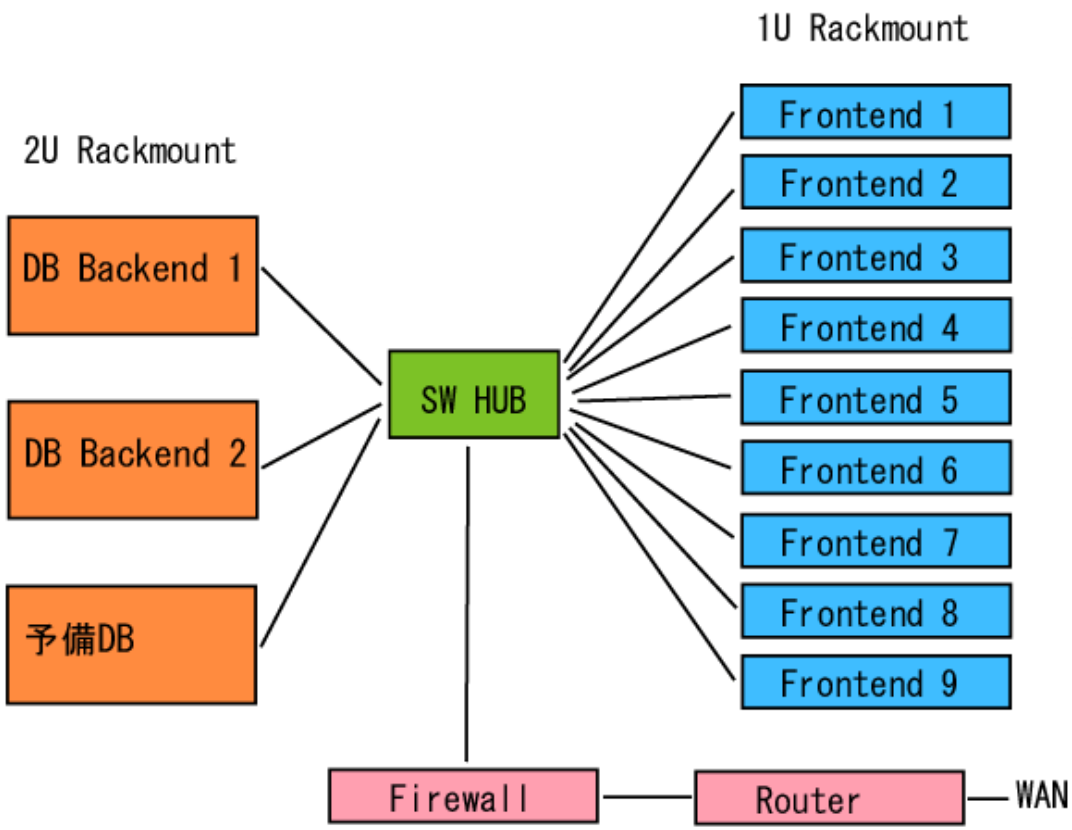
- ゲームは信頼性より中身重視（極論：動けばいいとも言う）
- 根本的に安くしたい
- ゲーム開発者はプログラミングスキルが高い
- 何が悪いのか、追及しやすい（かえって責任を果たせる）

オープンソースソフトウェアとゲームシステム開発の相性

相性のわるい点

- **OSSやUNIXを使ったことのないエンジニアが多い**
- **英語の壁**
- **GPLの問題**

ディプスファンタジアの物理接続図



＊Webは除く
＊フロントエンド1台あたり700常時接続をこなす

フロントエンドの特徴

* forkしないスタンドアロンサーバ
* 小さなクエリー（数十バイト）を専用プロトコルで送信
* CPU時間の90%以上をロジックが消費
* システムコールはネットワークのみ
* GCC + glibc + gdb で開発
* プログラムはクラッシュするものとして扱う

Dumb vs. Intelligent

Dumb なバックエンド

* 単機能
* 高速
* チューンしやすい
* 堅牢

Intelligent なバックエンド

* 多機能
* 高速にできるかも
* チューンしにくい
* 堅牢にできるかも

MySQLは、どっち？

テーブルの構造：極めて単純

Field	Type	Null	Key	Default	Extra
uid	varchar(32) binary		PRI		
gametype	int(11)		PRI	0	
slotid	int(11)		PRI	0	
version	int(11)	YES		NULL	
tag	varchar(128) binary	YES	MUL	NULL	
data	mediumblob	YES		NULL	
ord	bigint(20)		MUL	NULL	auto_increment

＊全てのテーブルを同じ構造にした

＊不要なものもいくつかあった

クエリーの内容:極めて単純

select data from charinfo where uid=X;

insert into charinfo values (X, Y , Z);

update from charinfo where uid=X and slotid=Z;

delete from charinfo where uid=X and slotid=Z;

アクセスパターン

書き込みのほうが、読み込みよりも多い
(insert,update)　　(select)

	読み ： 書き
最低	1 ： 1
ディプスファンタジア	1 ： 100
普通の検索エンジン等	100 ： 1

ベンチマーク時の性能(MyISAM)

4バイトのデータの場合(select)

SV CPU	CLI CPU	Network	Kernel	thr	max Q/s
P3 0.8	P3 0.8	100M	2.2.14	4	8783
P3 0.8	P3 0.8	1000M	2.2.14	5	8891
1900+	933x2	100M	2.4.9	4	12300
1900+	933x2	1000M	2.4.9	8	14820

40Kバイトのデータの場合(select)

P3 0.8	P3 0.8	100M	2.2.14	3	197
P3 0.8	P3 0.8	1000M	2.2.14	3	486
1900+	933x2	100M	2.4.9	3	216
1900+	933x2	1000M	2.4.9	4	771
1900+	933x2	1000M9k	2.4.9		計測不能

＊Update はselectの7割だった

実運用でピーク時の性能

40KBのデータを毎秒50～80回保存、5回読み込み
ベンチマークでは150回以上保存している。

＊ 1GHz P3×2 512M メモリのマシン

＊ bdflushやギガビットのチューンでもっと速くなりそう
→ ディスクの使い方に関するチューンパラメータがほしい

＊現在までMySQLが原因でクラッシュしたことはない

現場のエンジニアの意見

＊SQLは名前を知っているだけだったが、今回必要な分に関しては、すぐに覚えられた。
＊MySQLがなかなかインストールできなかった。
＊Linuxのコマンドを覚えるのが大変だった。
＊Emacsは嫌だがSambaで何とかなった。
＊MySQLは日本語のドキュメントがある方だったのでよかった
＊ゲーム作りついでにWeb掲示板も作ってみた。

コミュニティエンジンでの経験

＊TCP_NODELAYの問題などは、ソースがないと
なかなか分析・解決できなかった。

＊mediumblob　の失敗　→　チューン

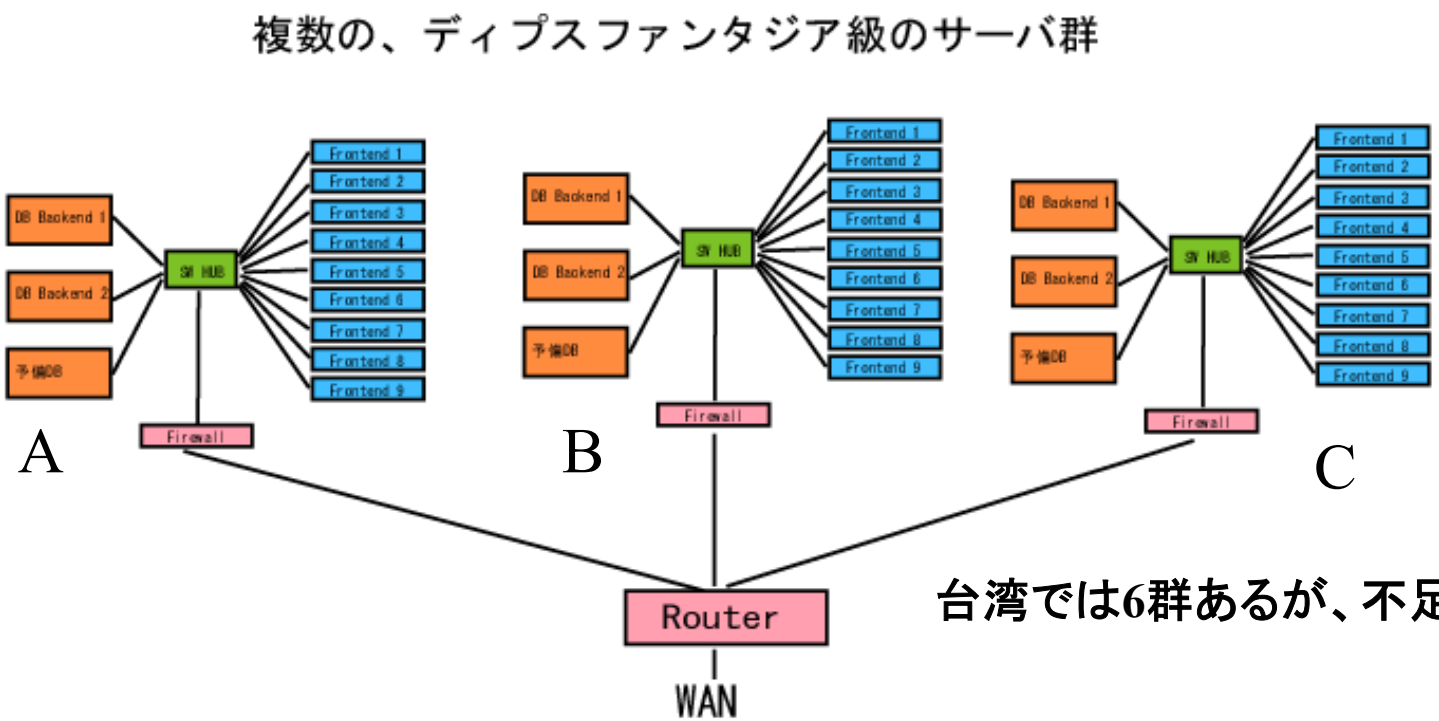
＊MySQLチューンよりカーネルチューンのほうが面倒。
多くの時間を、ディスクやメモリのチューン、ギガビット
のチューンに取られる。

＊MySQLはマシン性能から予想される最高性能が出る

結論。

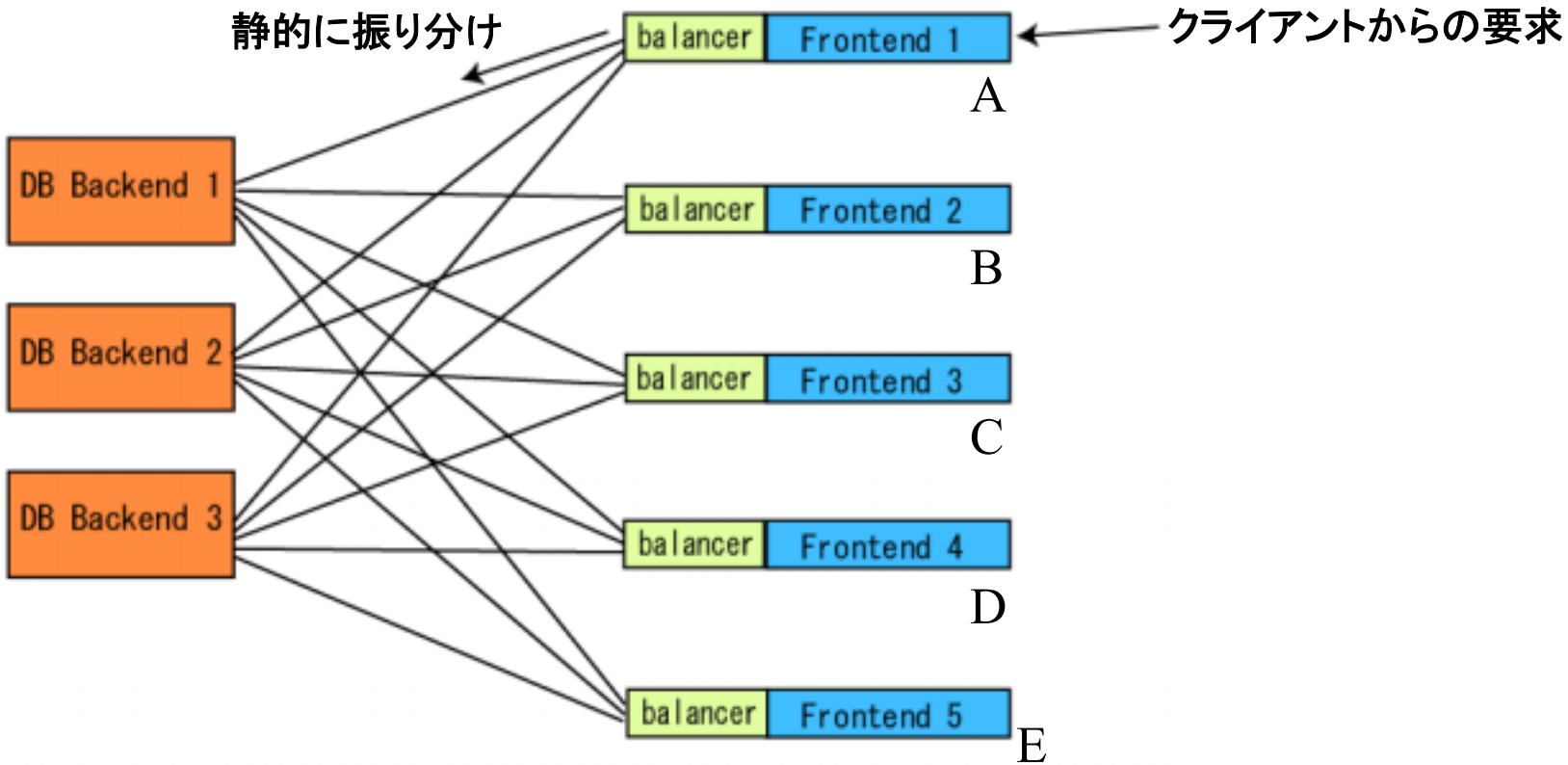
MySQLを使うと、オンラインゲームサービス用の
システム製作で要求される価格対性能比を実現できる。
ただしそのためには、C言語や英語など、OSSを使い
こなすためのスキルが若干は必要。

拡張性:単純な分割



台湾では6群あるが、不足している

拡張性 : 静的balancing（論理図）



フロントエンドに付けた　balancer が、ユニークキーに基づいて静的に振りわける

ゲームシステム開発　今後の課題

* 静的 balancing の実装と試験
* スクリプト言語を使った開発効率の向上
* 脱GPL化
* MySQL　への要望　（API、I/Oの使い方）(脱MySQL??)
* クラスターシステムのメンテナンスツール
* Gigabitなどの安くて速いデバイスでの実績

チュートリアルの内容公開URL

Http://www.vce-lab.net/osdb/

の予定（未完成）