クラウドの活用で大阪から世界へ。
チャットワークの挑戦

AWS Cloud Roadshow 2014 Osaka

Masaki Yamamoto

chatwork

自己紹介

ChatWork株式会社

専務取締役CTO 山本 正喜

2000年 兄とともに株式会社EC studioを学生起業

2011年 チャットワークを企画開発

2012年 ChatWork株式会社に社名変更

大阪府吹田市発

安定事業を無償譲渡。
シリコンバレーに乗り込んだ
起業家の挑戦。

チャットワーク
ChatWork株式会社

山本　敏行 氏 代表取締役

チャットワークとは

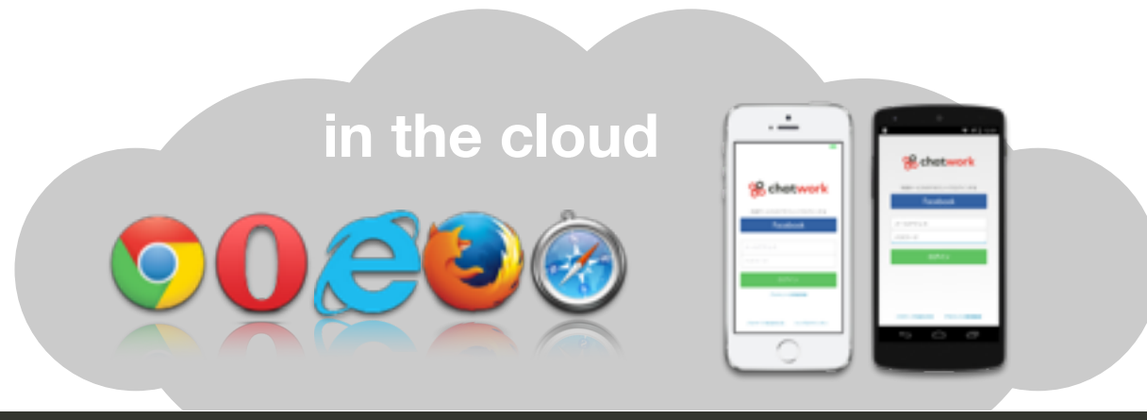
クラウド型ビジネスチャットツール

チャットの効率性・シンプルさをビジネスへ

チャット　　タスク管理

+

ビデオ通話



chatwork

導入企業数 5万1千社 突破！

株式会社 武蔵野　TEDx Kyoto　Lancers　KDDI　Funai Soken　HOME'S

KLab Cyscorpions　nanapi　ごちクル　CrowdWorks　Keio University

ZOHO Work・Online　京都大学 KYOTO UNIVERSITY　C-PIER　btrax　VASILY　CEREVO

YAMADA

1,000台のAndroid端末とともに導入　with KDDI

chatwork

本日お話すること

chatwork

Beyond chat. **Beyond simple**

社内システムからはじまったチャットワークが、どうクラウドを活用してスケールしていったか

チャットワークのはじまり

開発をはじめる前の状況

- ひとつ前につくっていたプロダクトが大失敗、、、
- 会社としてWebサービス自体に挫折感がただよう
- 会社としてはWebとリアルの融合という方向性に注力

チャットワークを思いつく

- 敗戦処理に追われる日々・・
- 何かやりたいなぁとチャットワークを企画したが、見事に却下
- どうしてもやりたい！と役員をひとりずつ呼び出し説得。

社長からの一言

「やってもいいけど、好きなことやるんやからひとりでやってや」

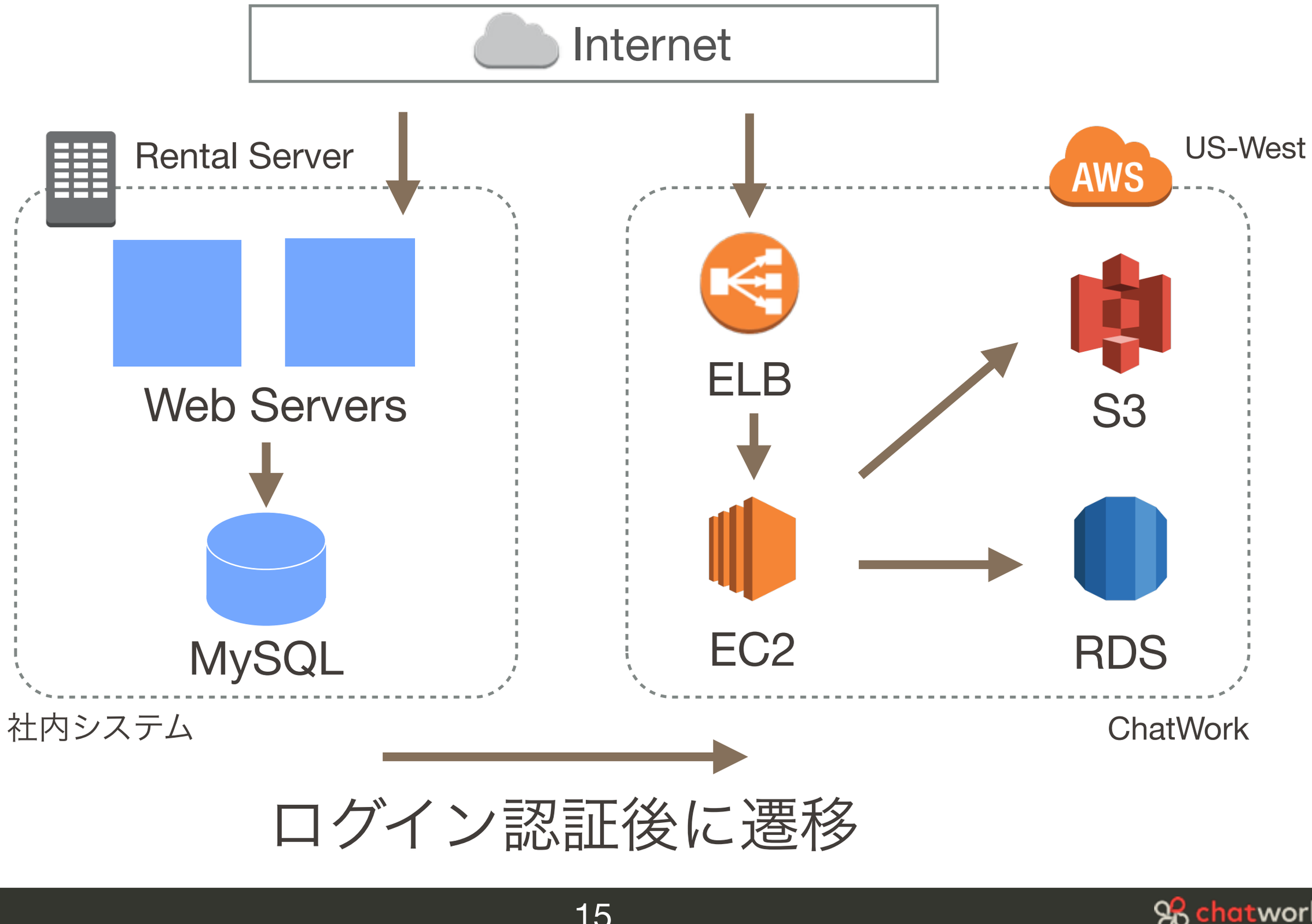
えっ

たったひとりからの開発スタート

- 社内の他スタッフからは何してるんだろう状態・・・
- フロントエンド、バックエンドなどもすべてひとりで開発。(デザインだけ少し手伝ってもらう

幻のファーストバージョン

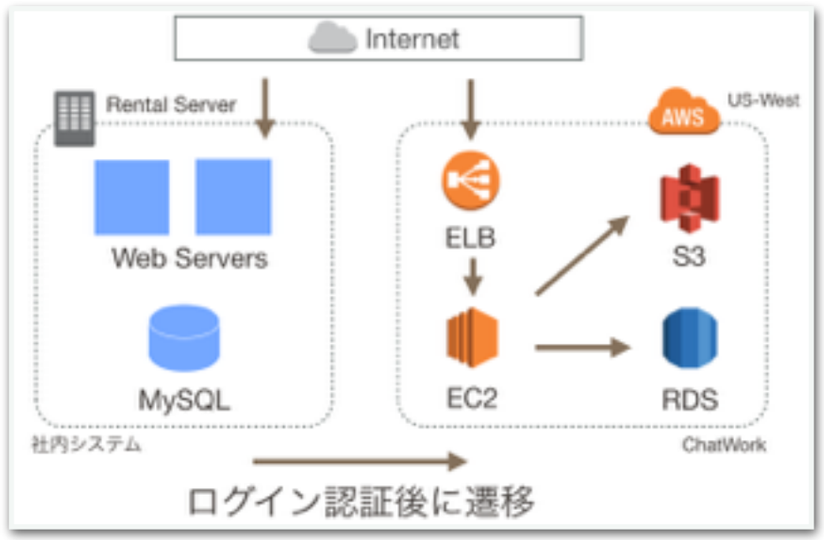
当時のサーバアーキテクチャ



ログイン認証後に遷移

部分的なクラウド導入

- 社内システムを基盤とすることで開発コストを最小化できた
- 今後のスケールも見据えて、新規開発の部分のみクラウドで



ビジネスとしての転機

- 社内ツールとしてそれなりに稼働してきたころ・・・
- 自社のUstreamチャンネルで社長がチラ見せしたところ大反響

社長からの一言

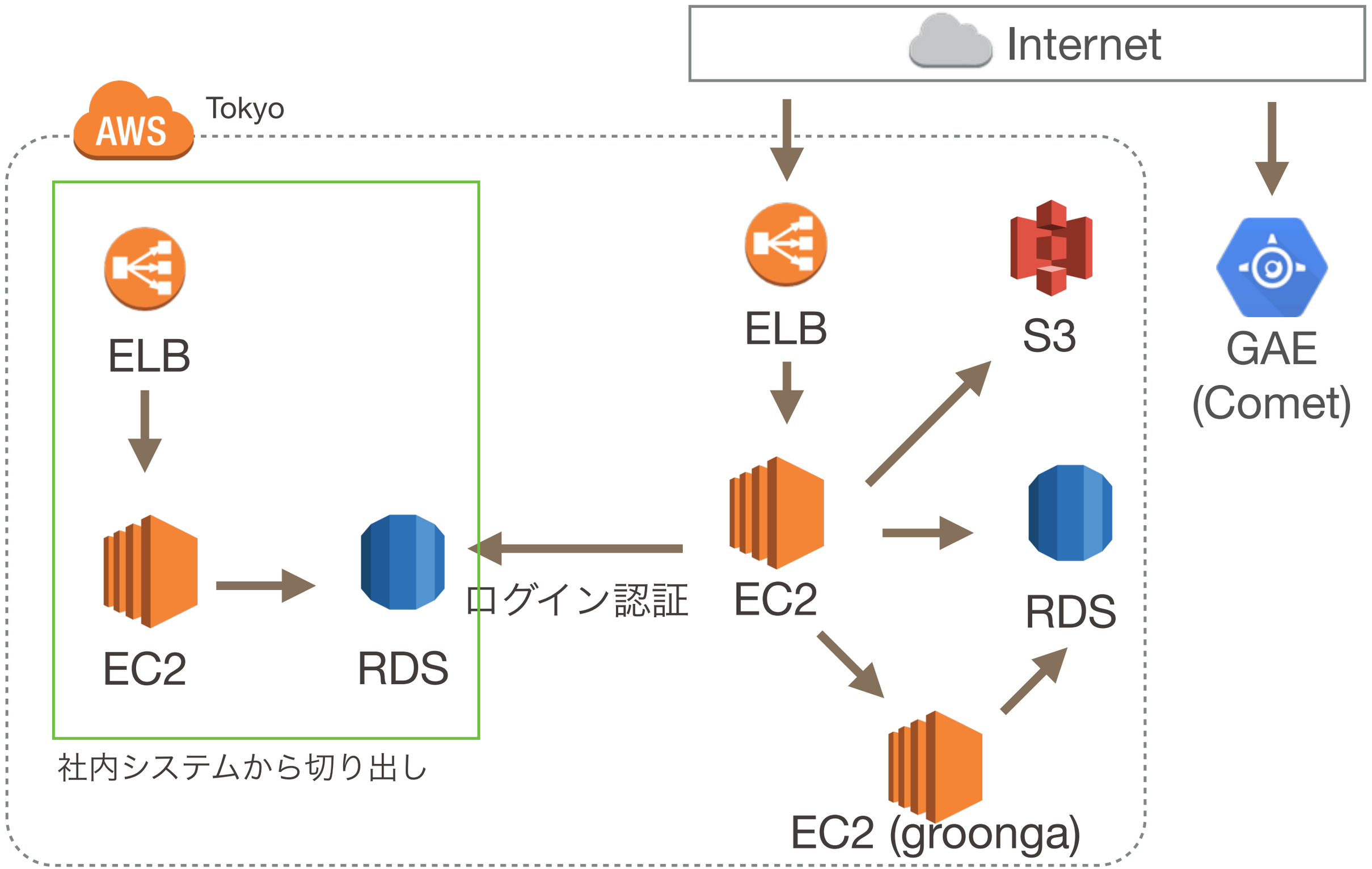
「これいけるで！商品化しよ！」

どうぞ！

事態は急展開

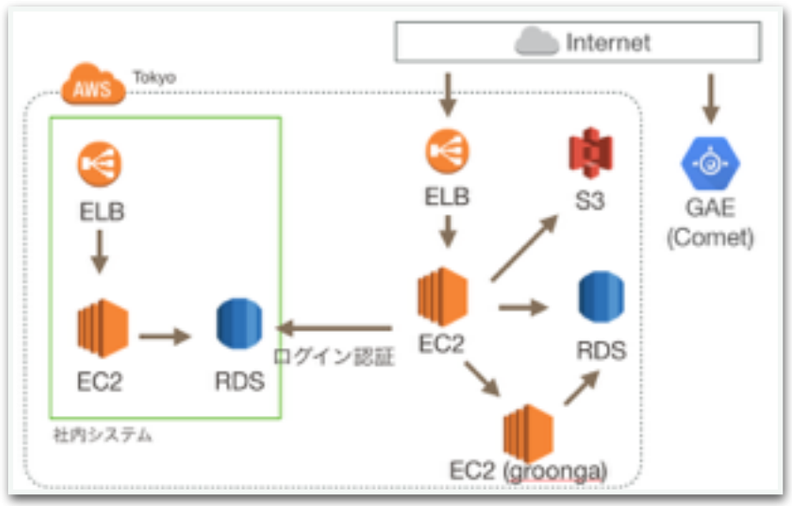
- 四人プロジェクトに昇格
- 正式なプロダクトとしてローンチ

当時のサーバアーキテクチャ



すべてのサーバをクラウドへ

- レンタルサーバ部分をそのままAWSへ移行。社内システム側と直接DB接続できるように。
- Tokyo Regionへ移行してレイテンシが大幅に改善。



Amazon RDSは最高のサービスです

- DBのスケールがとにかく課題。
  いざとなったらすぐにスケールアップできる心の余裕
- 何かあってもポイントインタイムリカバリで復旧できる安心感
  (不具合、オペミス)

Aurora楽しみですね

実感のこもったワンポイントアドバイス

- ELBは暖気申請しておきましょう
  (朝10時になると謎のダウン・・・)
- RDSのProvisioned IOPSは必須
  (ひっぱられます)
- クラウド(AWS)ならではのノウハウが必要。
  地雷を踏む前に詳しい人に聞くべし。

スケール、スケール、スケール

おかげ様でサービスは急成長 

- KDDI社と業務提携。エンタープライズのお客様へも多数導入。
- 社名をChatWork株式会社へ変更。社運を賭ける覚悟を決める。
- シリコンバレーにChatWork Incを設立。社長が移住して海外戦略強化。

しかし、、、

加速度的に増える負荷問題 

- ごく稀な条件がそろうと詰まって全体がサービスダウン
- 1組織でX,000人の大規模利用。裏のバックグラウンド処理が進まない、、
- ボトルネックが次々と顕在化

徹底的に負荷対策

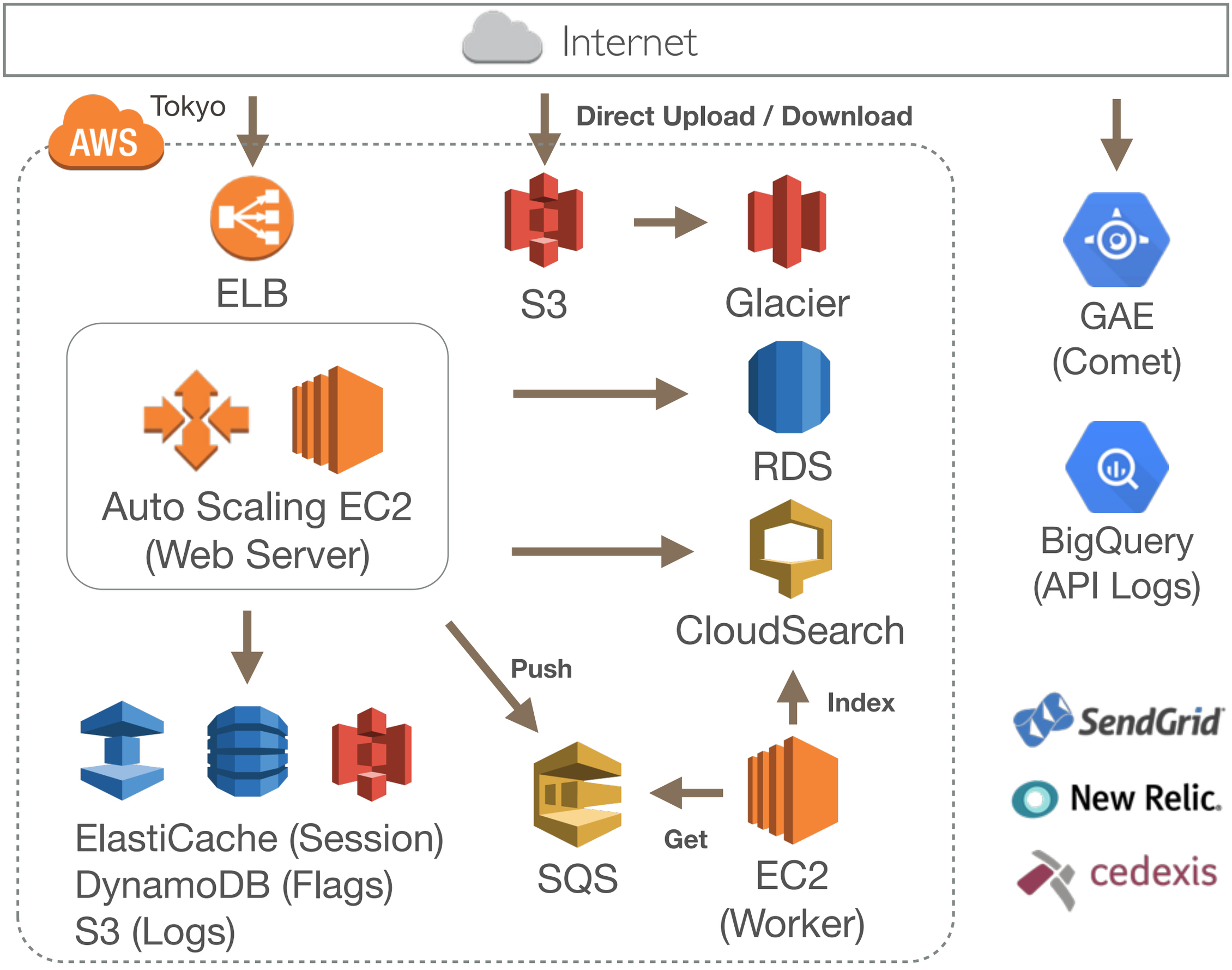
- SPoF(単一障害点)をひとつずつ地道につぶしていく作業。

→ 重い処理を疎結合化。SQSに投げて、バックグラウンドで非同期処理。

→ スケールアウトできるアーキテクチャへ移行。
DynamoDB、ElastiCache、CloudSearchなど。



現在のサーバアーキテクチャ


Internet
AWS
Tokyo
Direct Upload / Download
ELB
S3
Glacier
GAE
(Comet)
Auto Scaling EC2
(Web Server)
RDS
BigQuery
(API Logs)
CloudSearch
Push
Index
ElastiCache (Session)
DynamoDB (Flags)
S3 (Logs)
SQS
Get
EC2
(Worker)
SendGrid
New Relic.
cedexis

サービスの大幅安定化に成功

- AutoScalingの導入によりダウンタイムが大幅に短縮化
- 負荷特性に応じてスケールアウト型のAWSのマネージドサービスをフル活用。パフォーマンスが大きく上がった。

AutoScalingによるコスト削減

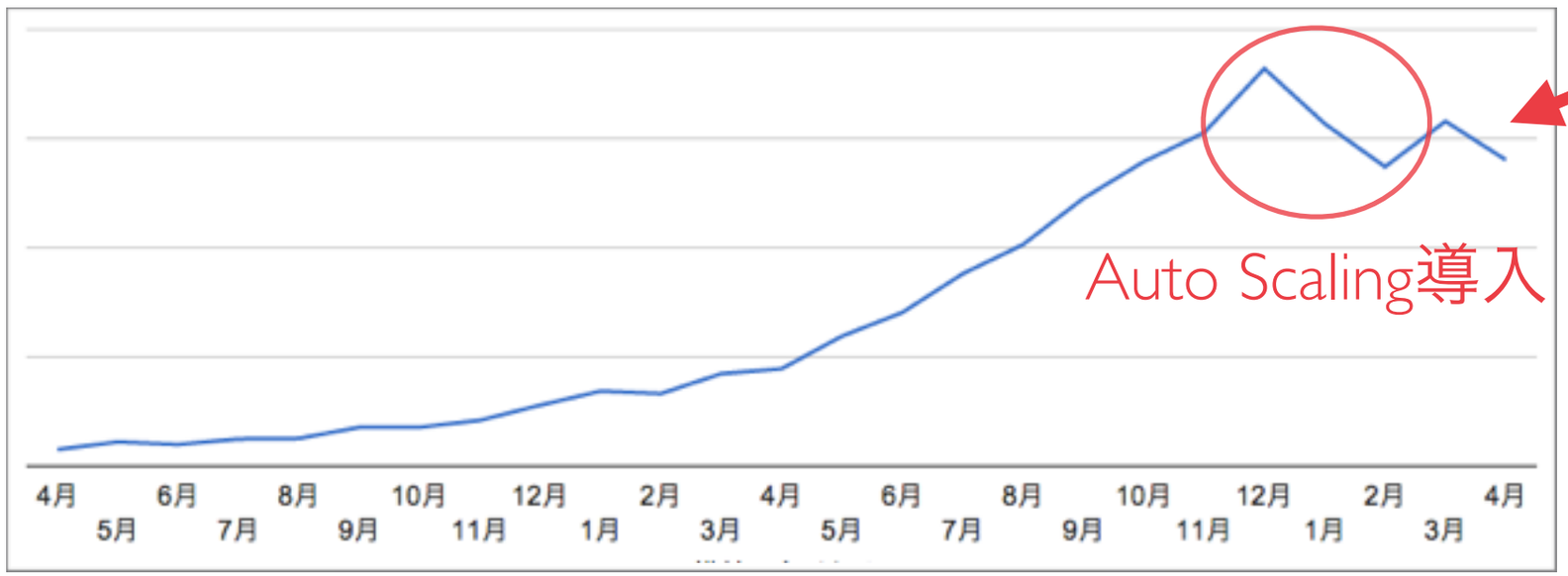
Throghputs

New Relic.

平日

休日

Costs



AWS値下げ

ありがたやありがたや

chatwork

現在の規模感 (2014年12月)

ユーザー	約 50万
メッセージ	約 4.7億
グループチャット	約 2,600万

amazon web services™

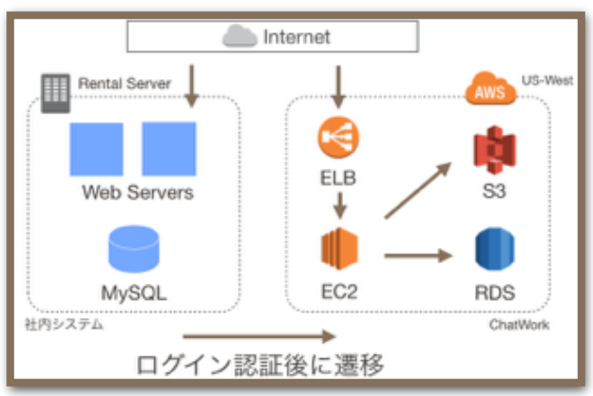
60+ EC2 Instances

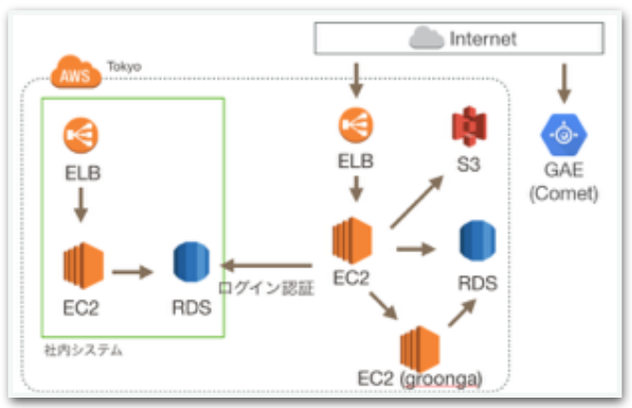
70TB+ S3/Glacier Storage used

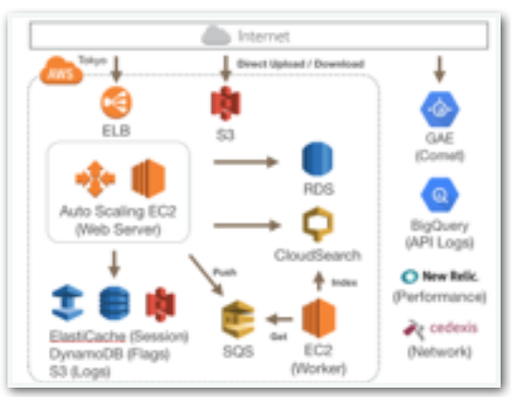
まとめ

アーキテクチャの移り変わり

初期開発期　　事業立ち上げ期　　事業成長期



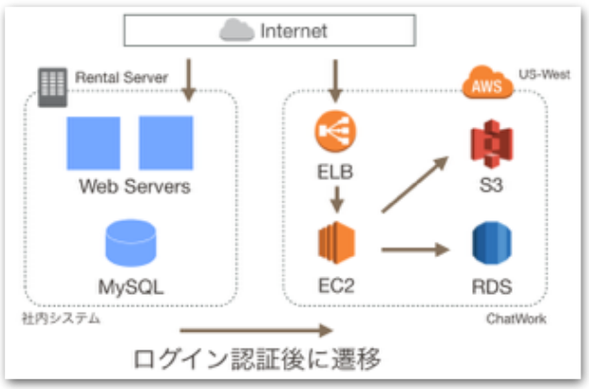


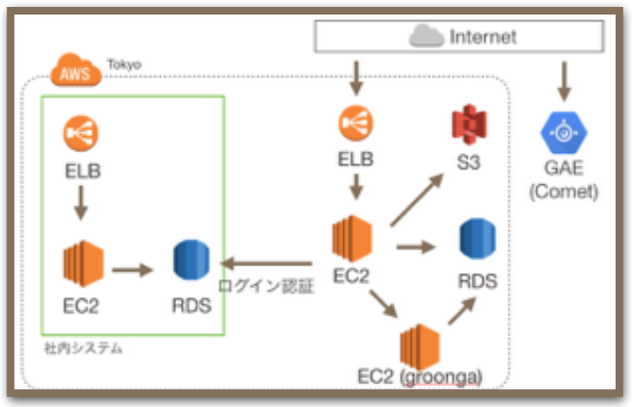


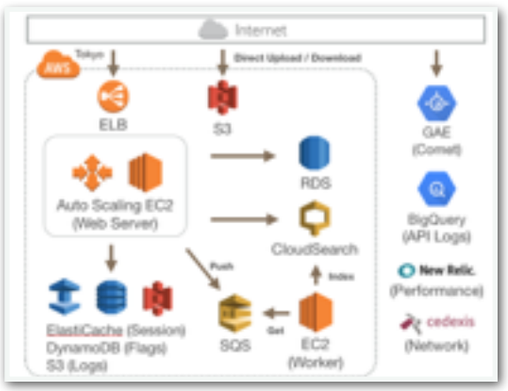
- 既存の社内システムを基盤として利用し、新規開発部分のみをAWSで外出しすることで開発コストを最小限に。

アーキテクチャの移り変わり

初期開発期　　**事業立ち上げ期**　　事業成長期



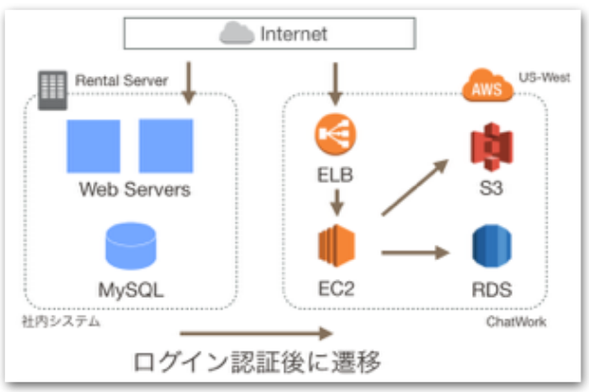




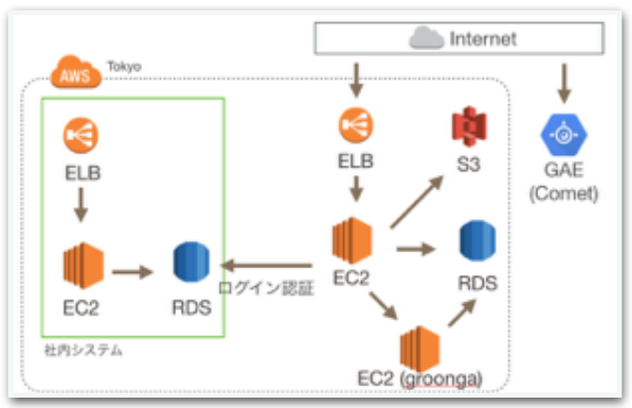
- すべてのサーバをAWSへ。AWSに統一することで異なるシステム間の連携が容易に。

アーキテクチャの移り変わり

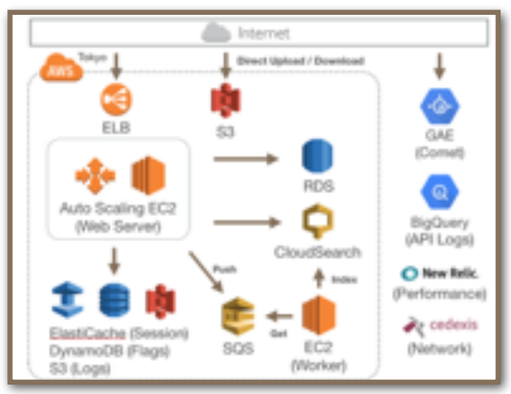
初期開発期



事業立ち上げ期



事業成長期



- 負荷対策のため、疎結合なアーキテクチャへ刷新。AWSにはマネージドサービスという形で多数用意されている。

立ち上げから拡大までAWSで

- 当初事業化できるかわからなかったチャットワークがここまで柔軟に成長できたのはAWSがあったから
- 成長してきてから壁にあたっても、AWSには解決策がある。
- できるだけ多くのAWSのサービスを使いこなすことが、スケールのカギ

大阪発、世界に向けてチャレンジしています

chatwork

Beyond chat. **Beyond simple**

ありがとうございました 🤲