

関西セミナー

気象庁XML利活用セッション

配信サーバを立ててみた

2013/07/11

クラウド・テクノロジー研究部会 リーダー

荒本道隆

(アドソル日進株式会社)

配信サーバを立てたワケ

- 気象庁からの配信はPubSubHubbub
 - グローバルIPを持ったサーバが必要
 - 受信用Subscriberの動作テストが面倒
 - いつ飛んでくるか分からない
 - Googleの動作確認ページの挙動がアヤシイ
- 会社を受信サーバを立ててみたけど
 - sshでログを見ないと、最新の情報が確認できない
 - 気軽に最新の防災情報XMLを見たい
- どうせやるなら
 - まだ使ったことのない技術を試してみたい
 - **そうだ、WebSocketで配信をしてみよう**

- **WebSocket** (ウェブソケット) は、[コンピュータ・ネットワーク](#)用の通信規格の1つである。[インターネット](#)の標準化団体である[W3C](#)と[IETF](#)が[ウェブサーバー](#)と[ウェブブラウザ](#)との間の通信のために規定を予定している[双方向通信](#)用の技術規格であり、[API](#)はW3Cが、WebSocket プロトコルはIETFが策定に関与している。プロトコルの仕様は [RFC 6455](#)。[TCP](#)上で動く。

Wikipediaより

- **Comet (Ajax) の代替として、策定された**

- **Comet** (コメット) とは、[Web アプリケーション](#)を構築する際に利用される技術で、この技術を使うと、[サーバ](#)で発生した[イベント](#)を[クライアント](#)からの要請なしに[クライアント](#)に送信することができる。
- Comet はこのような通信を実現するための複数の手法をまとめた概念である。これらの手法はブラウザにプラグインを追加することなく、([JavaScript](#) のような) デフォルトの機能で実現されるものである。理論的には Comet は、ブラウザがデータを要求する形の既存のウェブのモデルとは異なっている。実際は Comet アプリケーションは [Ajax](#) と [Long polling](#) を使用してサーバ上の新規データを取得する。

Wikipediaより

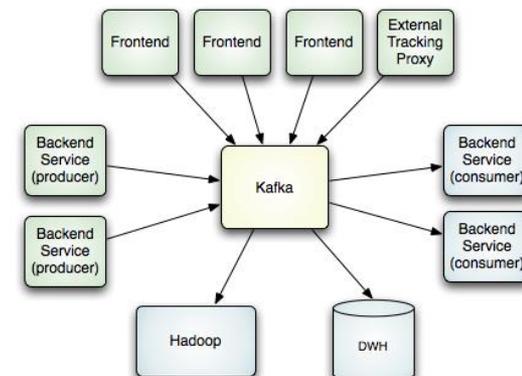
- **WebSocketの状況**

- HTML5から切り離された
- HTTP2.0は、対抗規格のSPDYが優位

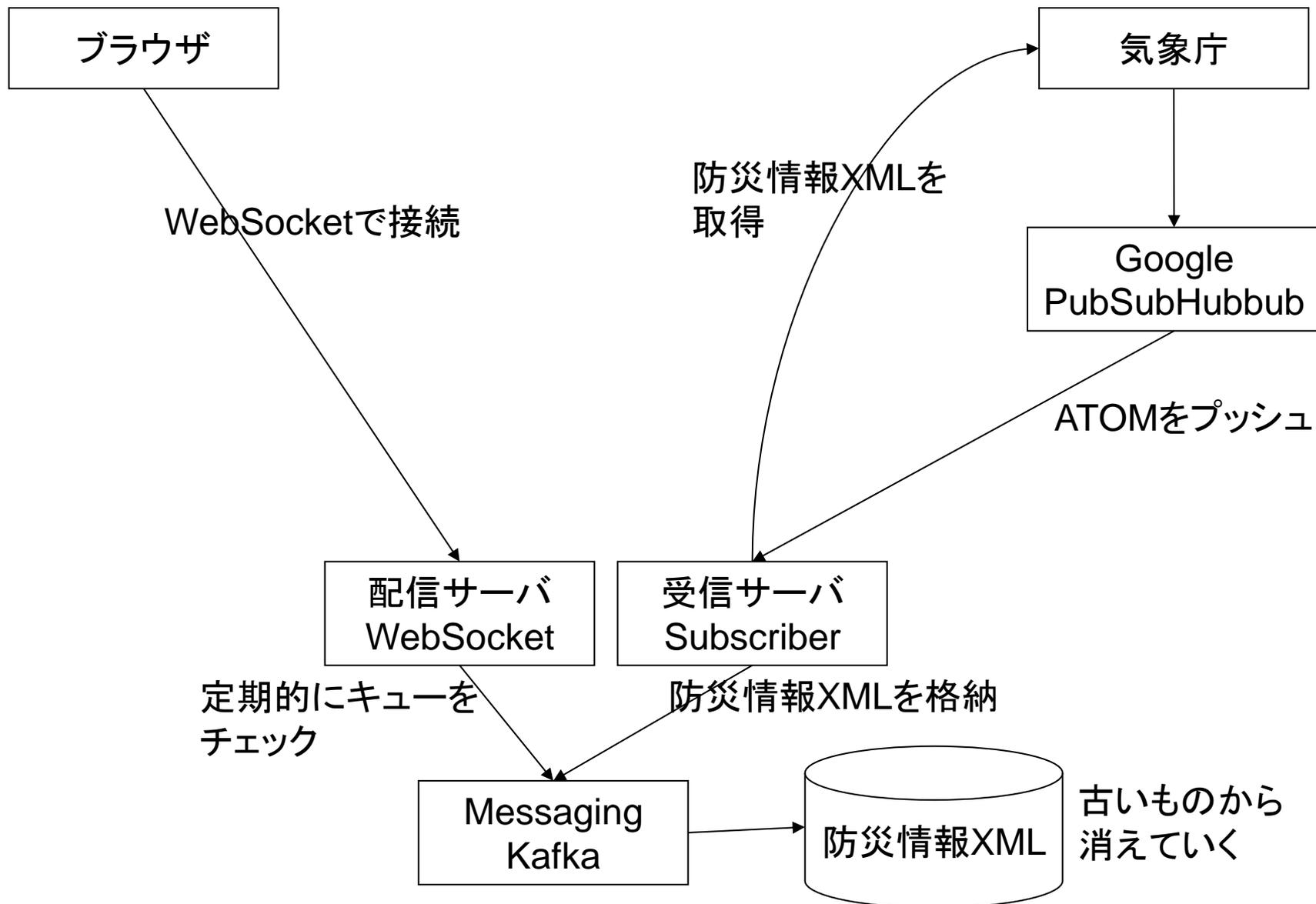
- サーバにはAmazonEC2を使用
- Subscriber
 - GoogleCodeからJavaのものを使用
 - <http://code.google.com/p/pubsubhubbub-java/>
 - Mainクラスを追加
 - PuSHhandler.javaに追記
 - `if (request.getMethod().equals("POST")) {`

```
public static void main(String[] args) {  
    PubSubHubbub.Web web  
        = new PubSubHubbub.Web(8888);  
    web.start();  
}
```

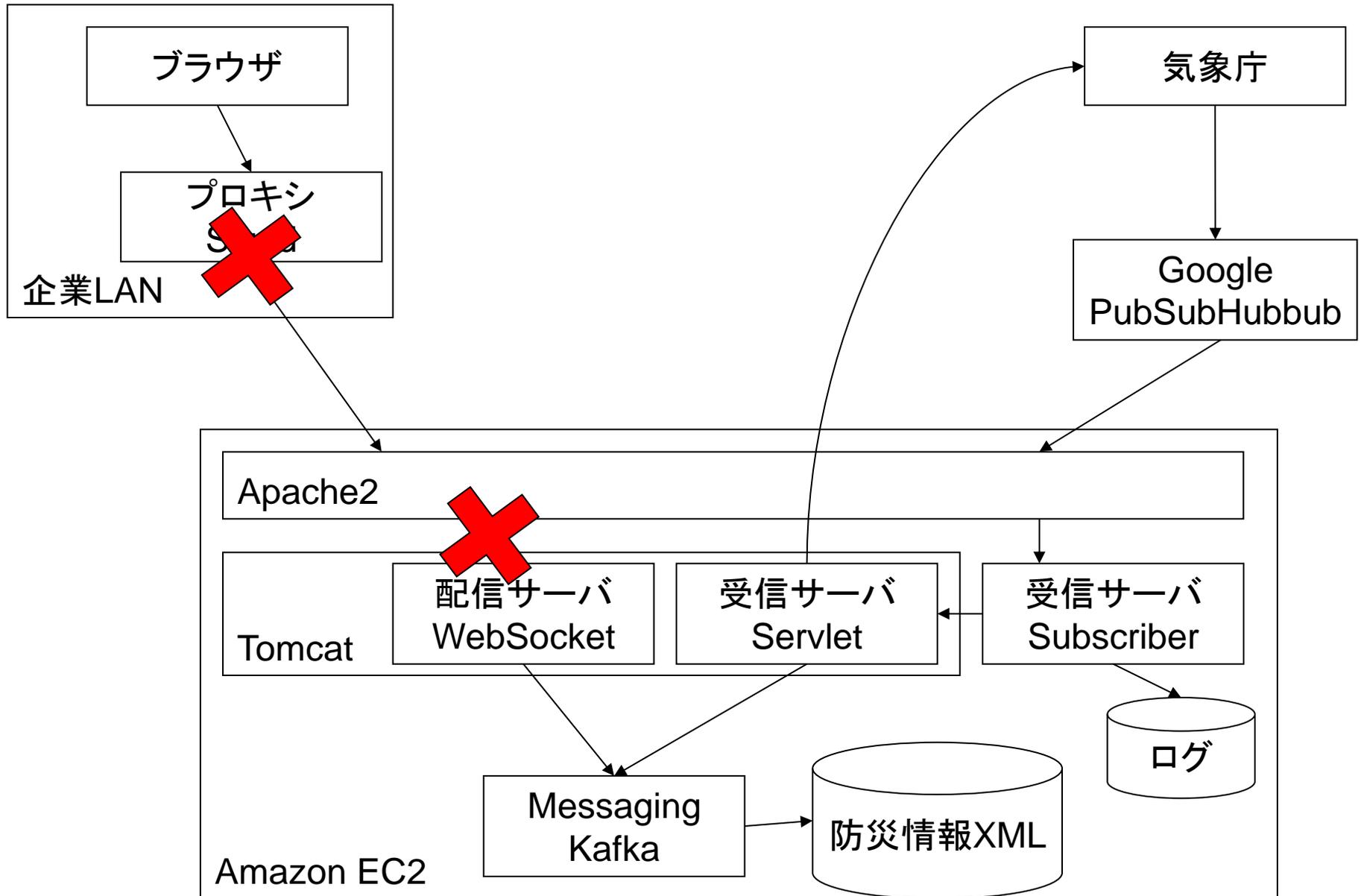
- WebSocket
 - Tomcat7のWebSocketServletを使用
 - `org.apache.catalina.websocket.WebSocketServlet`
- Kafka (messaging system)
 - SimpleConsumerを使用
 - 1つのメッセージを複数から取り出せる
 - Topic単位で古いものが消えていく



<http://kafka.apache.org/design.html>



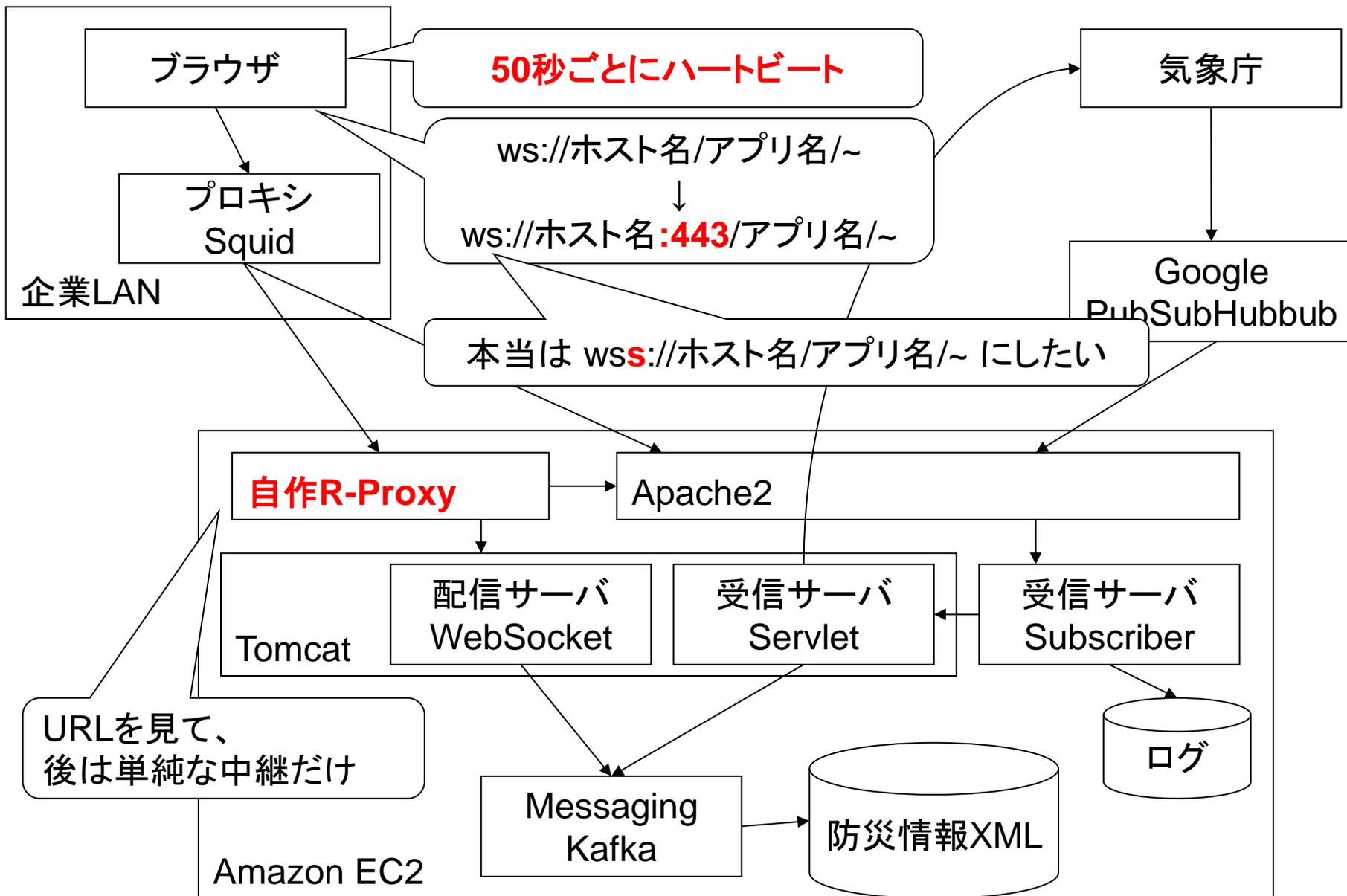
実際のシステム構成



WebSocket周りで次々と問題発生

- Squidがあると
 - ブラウザはCONNECTを使う
 - CONNECT cloud.projectla.jp:80 HTTP/1.1
 - Squidのデフォルト設定：**CONNECTは443以外を禁止**
- Apache2を入れると
 - セキュリティ確保や微調整をApache2でやりたい
 - 接続したままを想定していない
 - 「HTTP/1.1 101 Switching Protocols」で**子httpdが落ちる**
- Amazon EC2のロードバランサを入れると
 - **60秒でタイムアウト**するらしい

WebSocketの問題を解決



Kafkaでも問題発生

- 新規クライアントからのアクセス時
 - 全メッセージの先頭から順次空読み
 - offset値の最新50件だけをリストに残す
 - 最後まで読んだら、リストにある50件を返す
 - 以降、メッセージが増えれば、すぐにそれを返す
- Kafkaの特徴
 - 一定量のメッセージが溜まると、古い物から消す
- 2週間ほど連続運転してみた結果
 - 先頭(offset=0)が消去された
 - 先頭が無いので、**先頭から空読みができない**
 - メモリ上でキャッシュしていたので、Tomcat再起動時に発生
 - APIを調べても、「最後から読む」が無い？
- 解決方法
 - 「最新から50番目のoffset値」を常にファイルへ保存
 - 最初の空読みが不要になり、性能も改善

Comet版も作ってみた

- 作った理由
 - WebSocket版との比較
 - WebSocketが使えない環境があるかもしれない
- 性能
 - 最初の「直近50件を取得」は、WebSocketが圧倒的に早い
 - CPU負荷・ネットワーク負荷とも、WebSocketが優位
- 開発のし易さを比較
 - WebSocket版は作りやすかった
 - サーバ側は、永久ループ
 - クライアントへの配信エラーで、切断を検出
 - クライアント側は、イベントを待つだけ
 - ハートビートの書き込みエラーで、切断を検出
 - Comet版は色々と注意が必要だった
 - クッキーとパラメータを併用してoffset値を制御
 - Ajax呼び出し中に、前をキャンセルして、新しいリクエストを発行
 - タイミングによって発生するバグは、解決が困難

- WebSocket版
 - 接続して、コマンド文字を送信する
 - 「filter フィルタ文字列」
 - 「xpath パターン文字列」
 - 「start」
 - 50秒ごとにハートビート文字(空白文字)を送信すること
- Comet版
 - リクエストパラメータ
 - filer=フィルタ文字列
 - xpath=パターン文字列
 - offset=最初から取得したい時に0を指定
 - レスポンスでCookieに前回のoffset値をセット
 - 何もなければ、50秒ごとに空レスポンスが返る

サンプル画面紹介



Firefox

地方週間天気予報

AITC
先進IT活用推進コンソーシアム

気象庁防災情報XML受信ツール

WebSocket版

WebSocket+Kafkaにより、サーバ側で受信した防災情報XMLを配信します。
注意: Proxyを経由すると、WebSocketが接続できない場合があります。
注意: 50秒間隔でハートビートを行います。
注意: 受信した防災情報XMLをすべてブラウザのメモリ上に貯めるので、定期的に再接続

[Comet版はこちら](#)

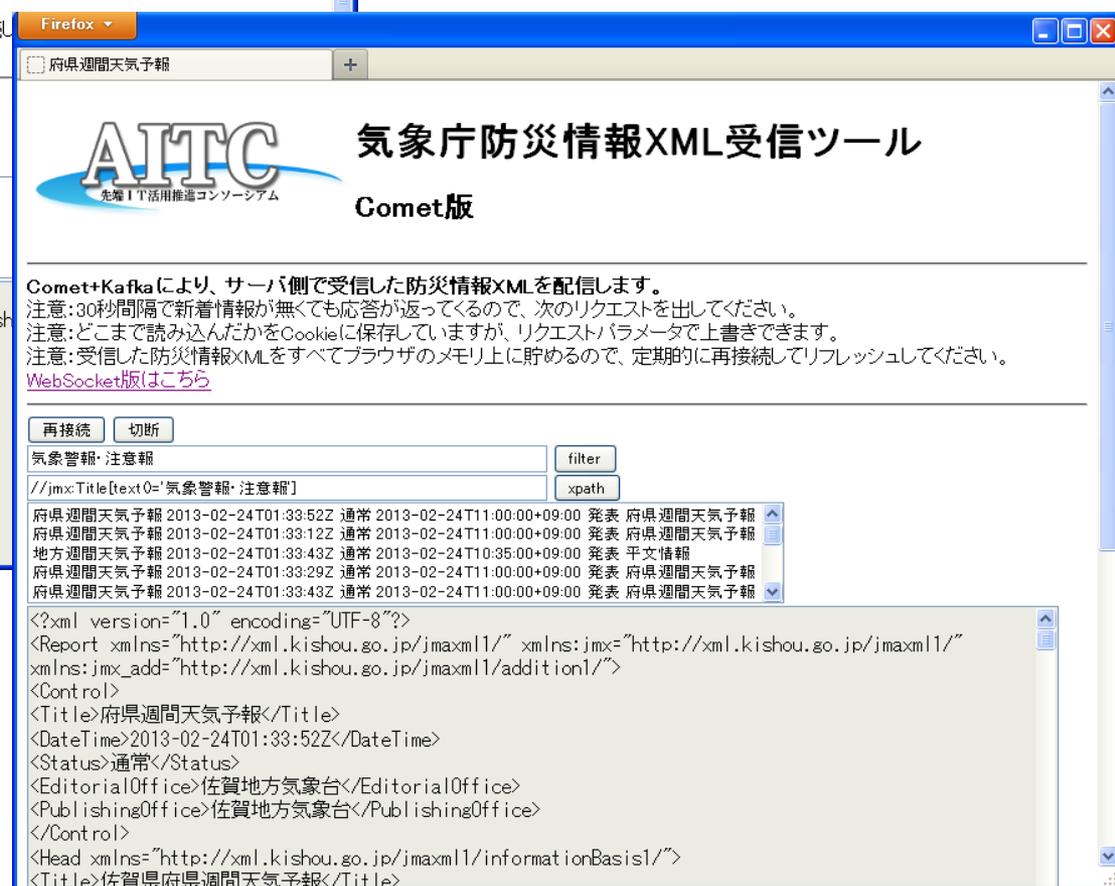
再接続 切断

気象警報・注意報

//jmx:Title[text 0='気象警報・注意報']

地方週間天気予報 2013-02-24T01:32:09Z 通常 2013-02-24T10:32:00+09:00 発表 平文情報
地方週間天気予報 2013-02-24T01:32:53Z 通常 2013-02-24T10:35:00+09:00 発表 平文情報
府県週間天気予報 2013-02-24T01:32:09Z 通常 2013-02-24T11:00:00+09:00 発表 府県週間天気予報
府県天気概況 2013-02-24T01:32:15Z 通常 2013-02-24T10:32:00+09:00 発表 平文情報
府県天気予報 2013-02-24T01:32:39Z 通常 2013-02-24T11:00:00+09:00 発表 府県天気予報

```
<?xml version="1.0" encoding="UTF-8"?>
<Report xmlns="http://xml.kishou.go.jp/jmaxml1/" xmlns:jmx="http://xml.kishou.go.jp/jmx/"
xmlns:jmx_add="http://xml.kishou.go.jp/jmaxml1/addition1/">
<Control>
<Title>地方週間天気予報</Title>
<DateTime>2013-02-24T01:32:09Z</DateTime>
<Status>通常</Status>
<EditorialOffice>大阪管区気象台</EditorialOffice>
<PublishingOffice>大阪管区気象台</PublishingOffice>
</Control>
<Head xmlns="http://xml.kishou.go.jp/jmaxml1/informationBasis1/">
<Title>近畿地方週間天気予報</Title>
```



Firefox

府県週間天気予報

AITC
先進IT活用推進コンソーシアム

気象庁防災情報XML受信ツール

Comet版

Comet+Kafkaにより、サーバ側で受信した防災情報XMLを配信します。
注意: 30秒間隔で新着情報が無くても応答が返ってくるので、次のリクエストを出してください。
注意: どこまで読み込んだかをCookieに保存していますが、リクエストパラメータで上書きできます。
注意: 受信した防災情報XMLをすべてブラウザのメモリ上に貯めるので、定期的に再接続してリフレッシュしてください。

[WebSocket版はこちら](#)

再接続 切断

気象警報・注意報

//jmx:Title[text 0='気象警報・注意報']

府県週間天気予報 2013-02-24T01:33:52Z 通常 2013-02-24T11:00:00+09:00 発表 府県週間天気予報
府県週間天気予報 2013-02-24T01:33:12Z 通常 2013-02-24T11:00:00+09:00 発表 府県週間天気予報
地方週間天気予報 2013-02-24T01:33:43Z 通常 2013-02-24T10:35:00+09:00 発表 平文情報
府県週間天気予報 2013-02-24T01:33:29Z 通常 2013-02-24T11:00:00+09:00 発表 府県週間天気予報
府県週間天気予報 2013-02-24T01:33:43Z 通常 2013-02-24T11:00:00+09:00 発表 府県週間天気予報

```
<?xml version="1.0" encoding="UTF-8"?>
<Report xmlns="http://xml.kishou.go.jp/jmaxml1/" xmlns:jmx="http://xml.kishou.go.jp/jmaxml1/"
xmlns:jmx_add="http://xml.kishou.go.jp/jmaxml1/addition1/">
<Control>
<Title>府県週間天気予報</Title>
<DateTime>2013-02-24T01:33:52Z</DateTime>
<Status>通常</Status>
<EditorialOffice>佐賀地方気象台</EditorialOffice>
<PublishingOffice>佐賀地方気象台</PublishingOffice>
</Control>
<Head xmlns="http://xml.kishou.go.jp/jmaxml1/informationBasis1/">
<Title>佐賀県府県週間天気予報</Title>
```

- クラウド（AmazonEC2）上にWebSocketサーバを構築
 - 予想外の問題がいろいろと起きた
 - まだWebSocketはあまり使われていない？
 - CPU負荷やネットワーク負荷を下げるために、利用価値は高い
 - WebSocketで社内LAN→クラウドの接続が確認できた
- ブラウザに対するPush通知が実現
 - 最初は、ブラウザを開いておいて眺めていたけど
 - 通知機能が欲しい
 - 関連する地域名が含まれていたら...
 - 緊急度の高いXMLが届いたら...

- 防災情報XML**試用**のために利用してください
 - 本番システムには組み込まないでください
 - 突然、サービス停止することもあります
 - 提供期間
 - AITCクラウド・テクノロジー研究部会が存続する間を予定
 - メンテナンス
 - 不定期に行います
- 蓄積サーバの検索ページ
 - AITC会員以外も利用OK
 - <http://cloud.projectla.jp/jmaxmlldb/>
- 蓄積サーバのクエリページ、配信サーバ
 - AITC会員限定
 - URLは staff@aitc.jp にお問い合わせください