

# WebSocketを用いたEPICS Channel Access Clientの開発と実装

総研大

KEK

理研仁科

内山 暁仁

古川 和朗

日暮祥英

# Webを用いたシステムのメリット

- ✓ WANを利用したシステム提供
- ✓ オペレーションの場所の制約が無くなる
- ✓ 専用ソフトウェアが必要なくなる(ブラウザを利用)
- ✓ GUI開発の容易さ
- ✓ ブラウザが実装されている機器はスマートフォンやPDA色々ある



# HTTP(HyperText Transfer Protocol)

コネクションを張ってから切断(disconnect)するまでの一連の動作

CGI  
Webアプリ

GET /sample1/Hello.cgi?message=ABCD  
POST/sample1/Hello

HTTP Request

HTML  
HTTP Response

サーバサイド

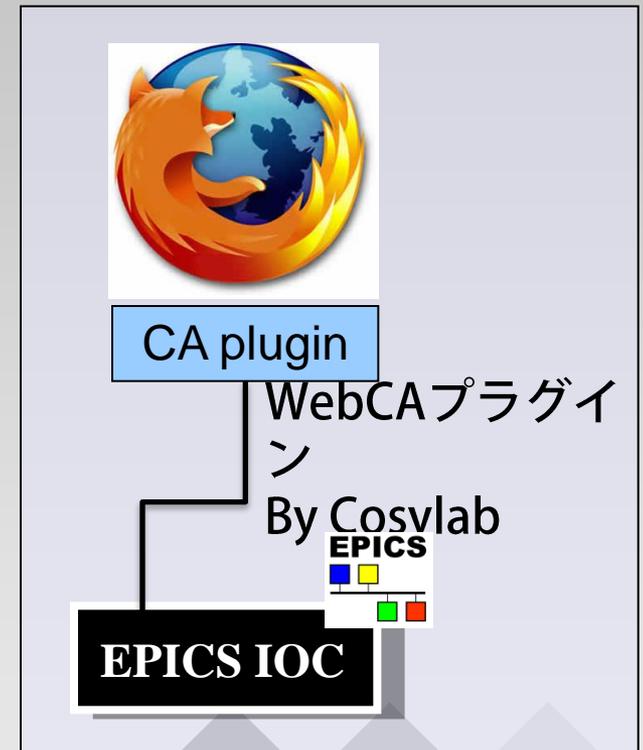
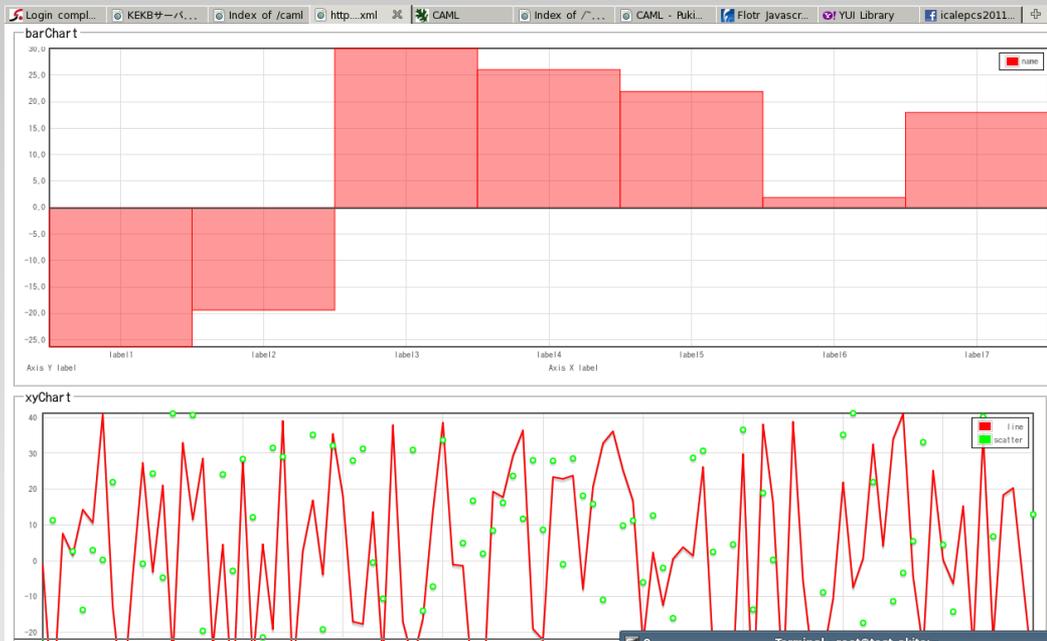
クライアントサイド



# EPICSにおけるWebブラウザからの制御(1)

## CAML

Channel Access Markup Language by SNS



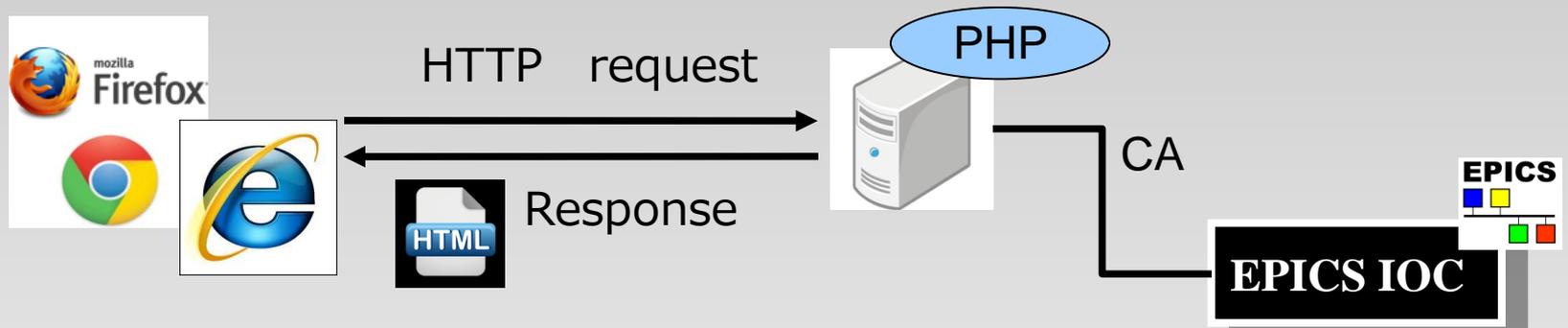
- ブラウザに依存
- 直接Channel AccessするのでLANやPC以外での利用は困難
- サンプルでは 1 sec

# EPICSにおけるWebブラウザからの制御(2)

## PHP EPICS Module

by PSI

- PHPから**Channel Access**を呼べるモジュール

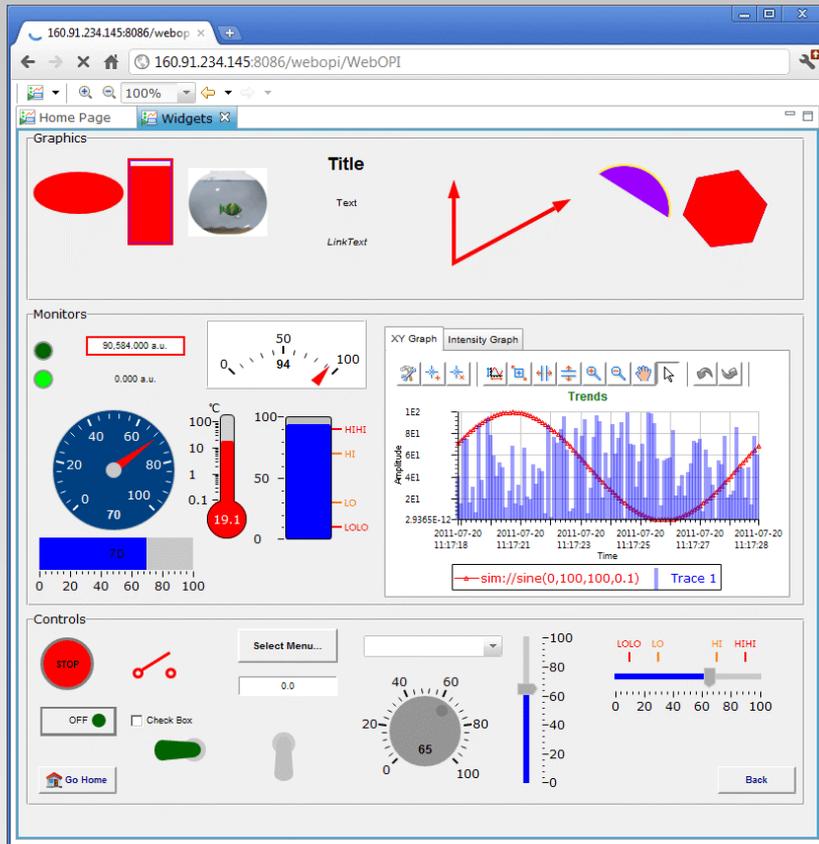


- サポートは止まっている(Only PHP4)
- CAのAPIの提供にとどまるのでウェブアプリは書く必要がある
- CAのモニタ機能 (イベント・ドリブンな通知) の利用が困難
- **Ajax**を使いある程度(イントラネットで約50 msec)の周期でアクセスを確認

# EPICSにおけるWebブラウザからの制御(3)

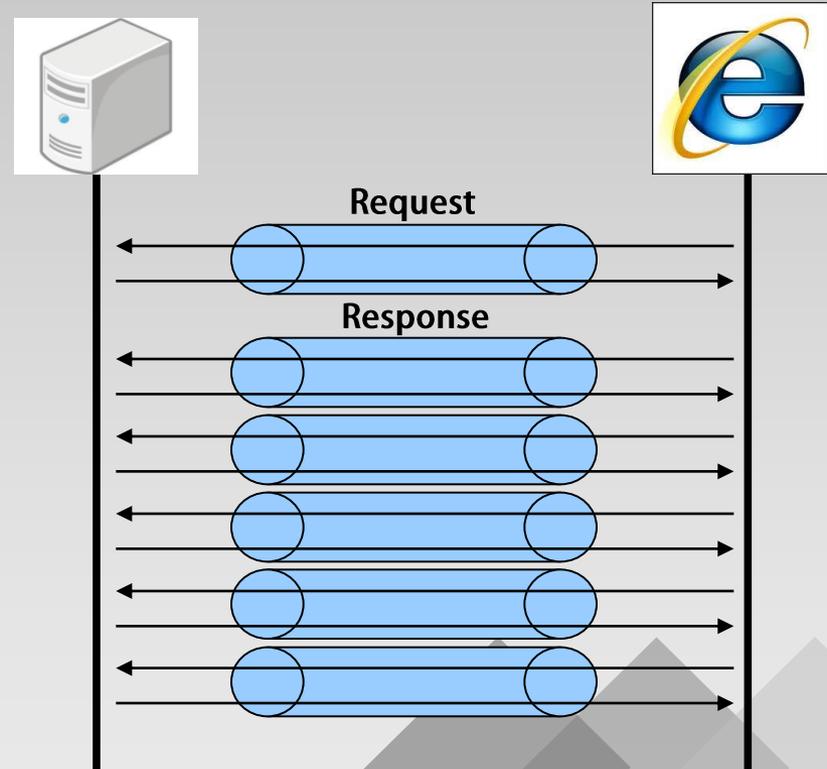
## WebOPI by SNS

- ・ CSS/BOYで開発したOPIファイルをWebで提供可
- 要Tomcat 7



## Ajax

(Asynchronous JavaScript + XML)

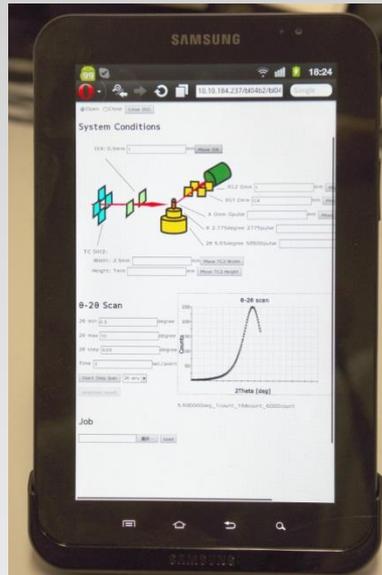
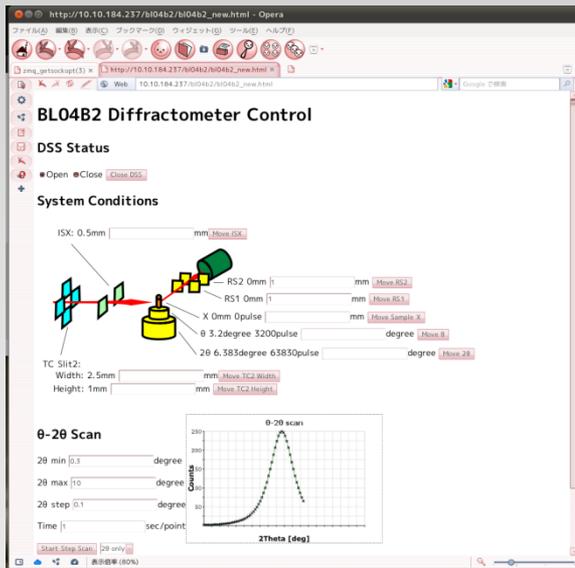


- ・ Ajax通信を利用してインタラクティブなWebを提供
- ・ イン트라ネットな環境では 5 msec ~ 100 msec でアクセス

# 加速器制御におけるWebSocketの可能性

国際会議ICALEPCS2011@フランス グルノーブル 2011/10

- 古川行人 氏@SPRING-8がビームライン制御 (MADOCA) におけるWebSocketの有用性を発表
- シンプルなCで書いたWebSocketサーバをプロトタイプで構築・動作確認



Reference: Proc. ICALEPCS2011, No.WEMAU10, Y. Furukawa

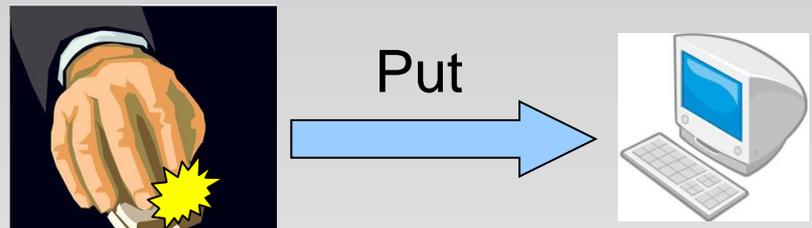
# WebSocket ( 1 )

## ◆リアルタイムに双方向通信を行う為のプロトコル

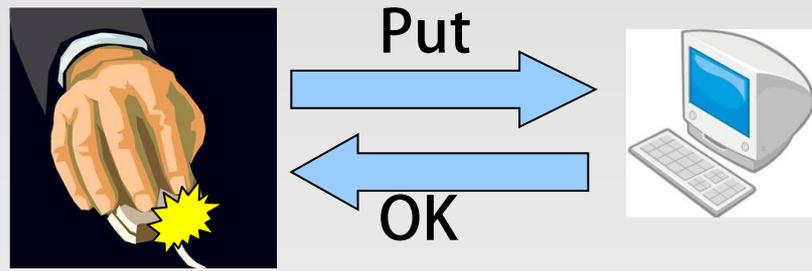
2011/12仕様決定

### Ajaxの弱点

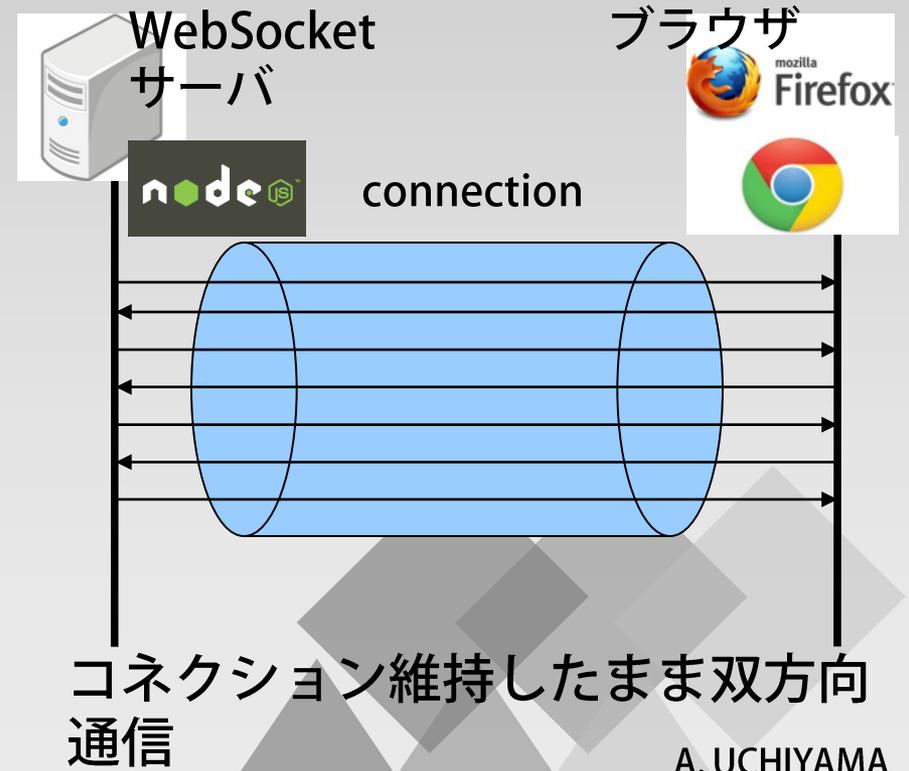
- 通信の都度コネクションを貼らなければならない
  - インタラクティブ化の実現には命令を送ってからの応答速度が重要
- 送った命令に対する確認の手段がない (Callbackがない)
- クライアントがサーバの状態を判断できない



Put ... 成功？失敗？



CallbackにOKが返ってきたから成功。



# WebSocket (2)

## デメリット

現在全てのブラウザが対応しているわけではない



Mozilla Firefox



Google Chrome



Opera



Internet Explorer 9



IE10からWebSocketに対応する事が決定



標準化

# Requirement

EPICSで実現するためには・・・

1. Channel Accessに対応するWebSocketサーバの開発
  - ✓ **Node.js** (サーバサイドJavaScript言語) を利用
2. Node.jsからChannel Access Protocolを呼び出す機能を実装する
  - ✓ C++アドオンで提供 (**Channel Access for Node.js**)
3. クライアントの実装
  - ✓ 可視化(グラフ・チャート)に向けてJavaScriptライブラリの選択
  - ✓ テキストアップデートはDOM操作

## 特徴

簡単にネットワーク・プログラムが書ける

サーバサイドJavaScript (通常JavaScriptはブラウザで動作する言語)

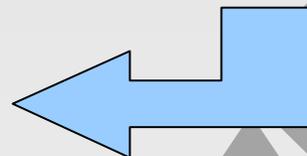
- ✓ 全部C言語などで実装する事は難易度が高い
- ✓ クライアントもサーバーサイドもJavaScriptで書く事が可能

Node自身はGoogle V8 JavaScript Engineを元に開発

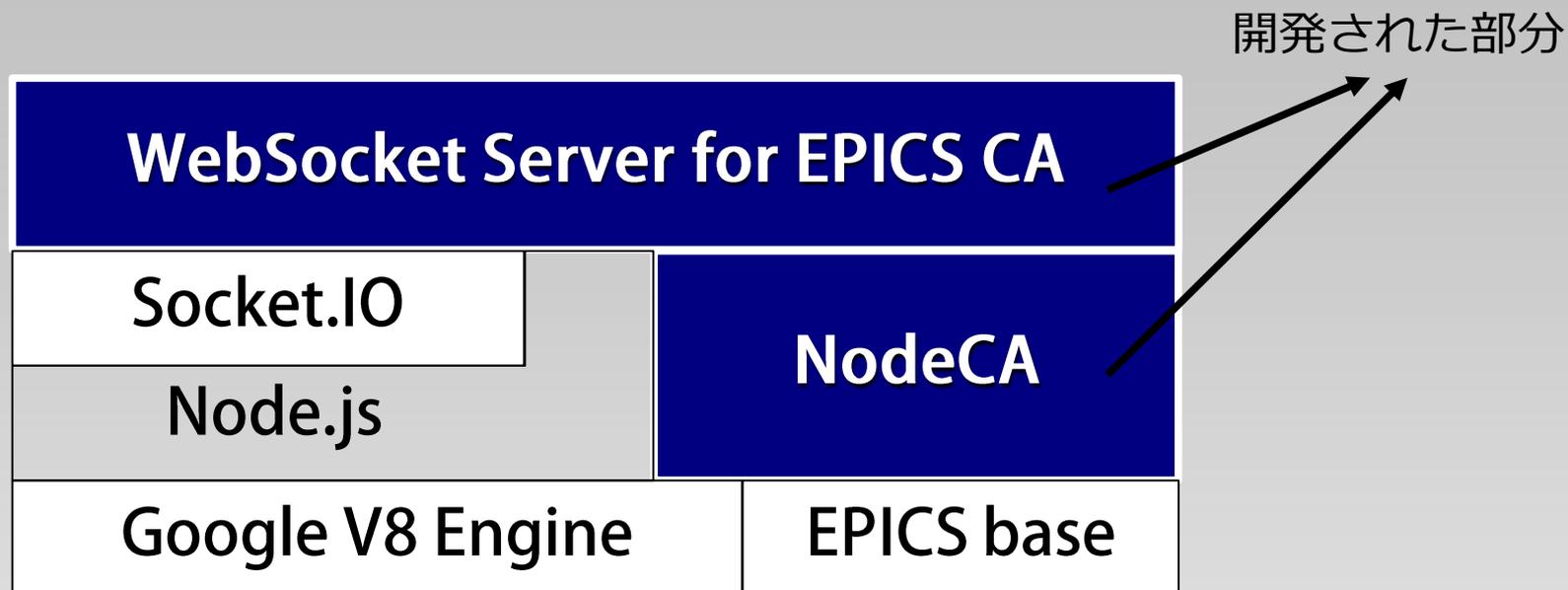
- ✓ Google V8 EngineはChromeでJavaScriptを実装する為に開発
- ✓ 新しい言語であるためAPIの仕様が変わる可能性がある

WebSocketのAPIを提供するライブラリがある

Socket.IO



# NodeCA・WebSocketサーバ概要



caget Example(サーバサイド)

```
var pv = require("../build/Release/nodeca");  
val = pv.caget("akito12:example");  
console.log(val);
```

アドオン読み込み

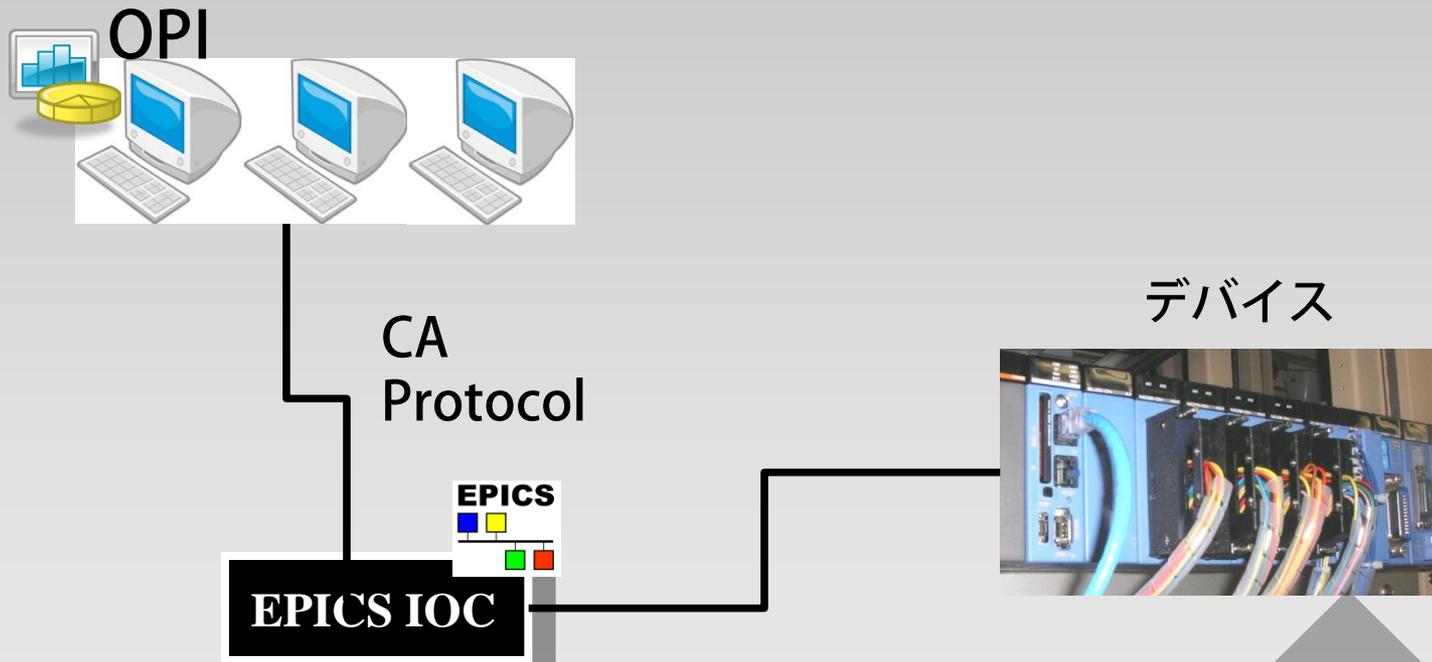
EPICSレコード指定

出力

コンセプトはできるだけシンプルに  
プログラムが書ける事

# システム概要

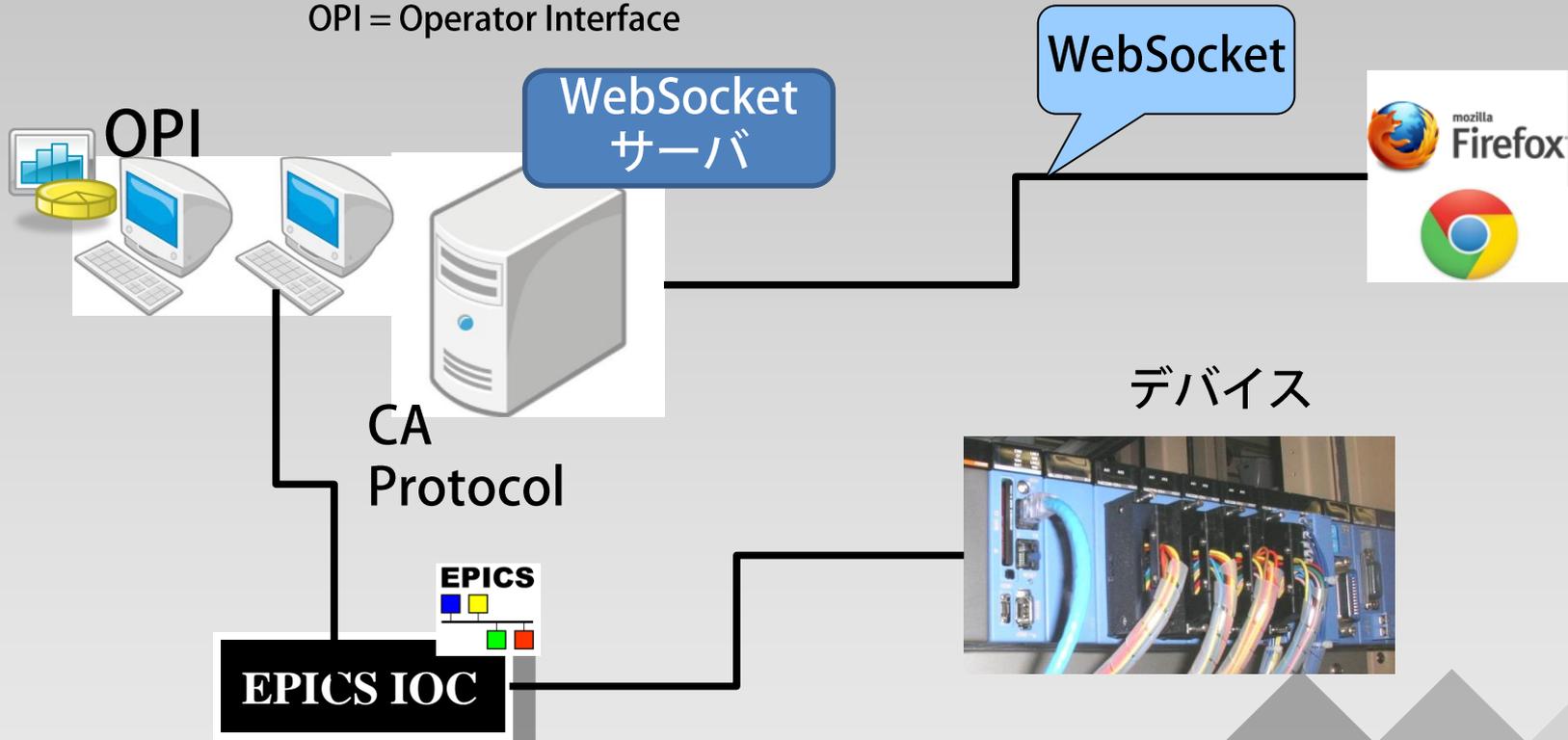
## 通常のEPICSシステムの例



# システム概要

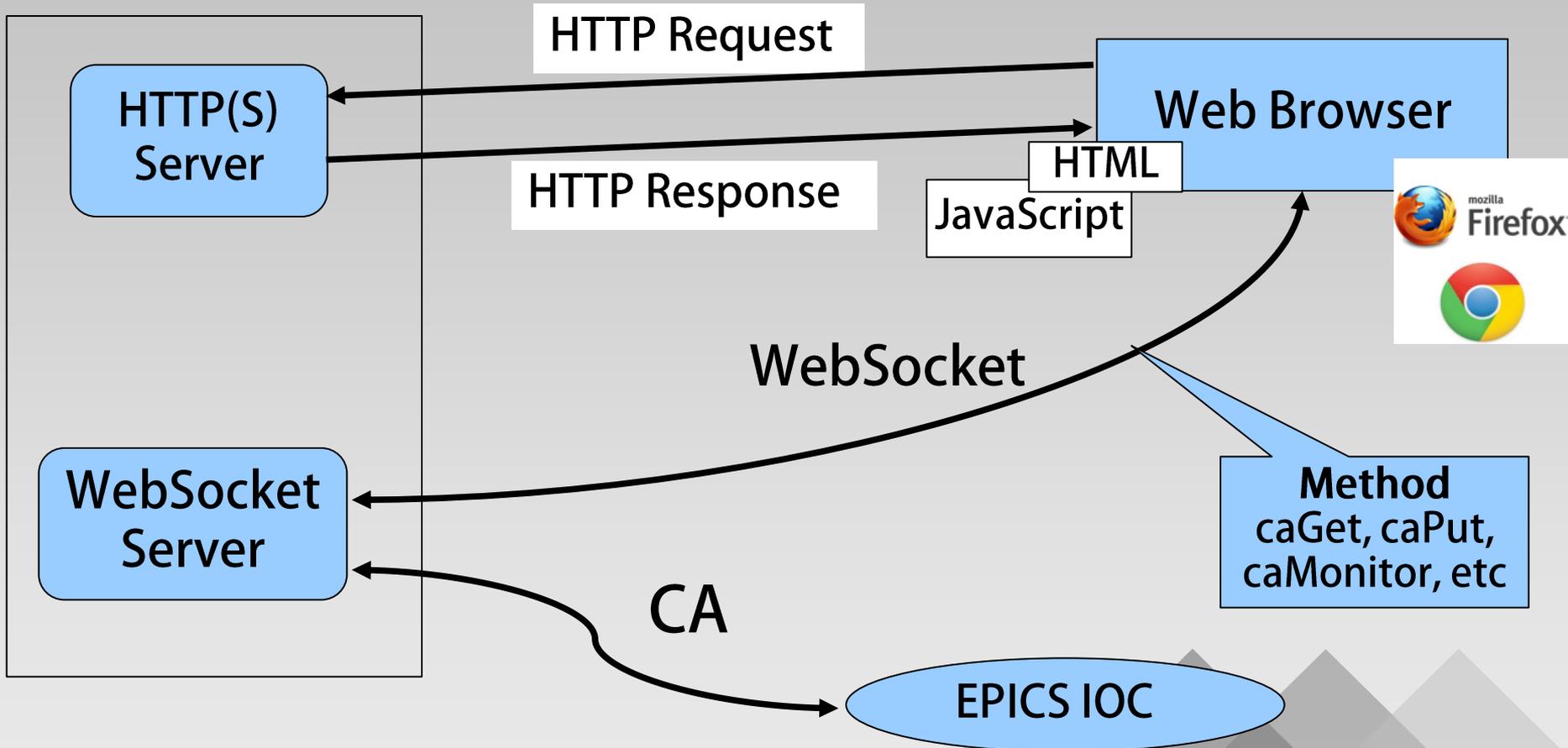
## OPIとWebSocketサーバが同列

OPI = Operator Interface



# WebSocket概要

- 本開発ではWebSocketサーバとWebサーバを一緒に提供（同じポート）



# クライアントサイド実装(JavaScript)

```
/* 送信 */  
var socket = io.connect('/ca/gauge');  
socket.emit('caMonitor',{  
  pv: 'akito12Host:ai1',  
  data_name: 'ai'  
})
```

EPICSレコード

データ名を指定

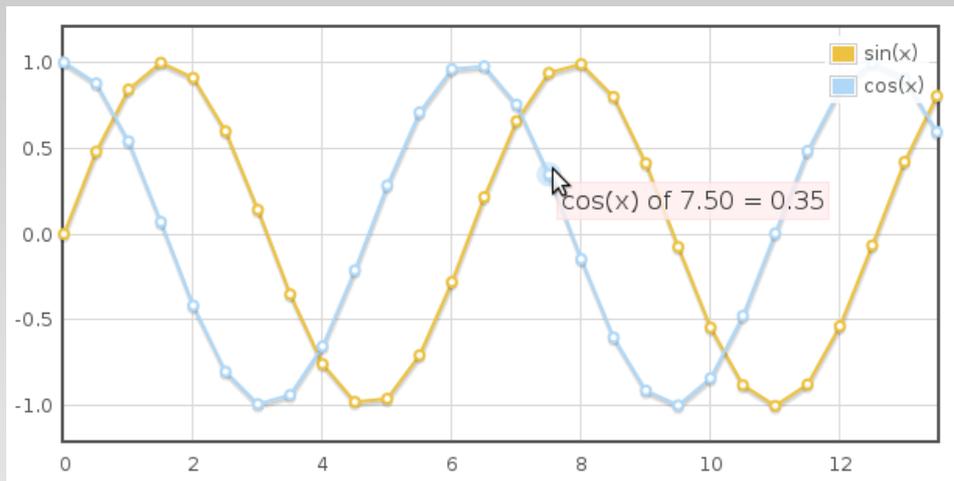
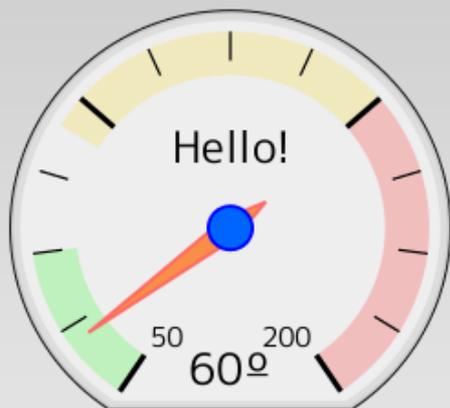
```
/* 受信 */  
socket.on('ca', function(data){  
  var value = data.ai;  
});
```

データ受信

# クライアントサイド実装(可視化)

リアルタイムなデータを可視化するためJavaScriptライブラリを利用

- jsgauge
- flot (jQueryベース, MITライセンス)



見栄えのするチャートをゼロから作るコストは大きい。  
他の有料なチャートライブラリの使用も。

OPI用GUI開発 ≒ Web(HTML)開発

難易度は大幅に下がる

# クライアントサイド実装デモ

Firefox

信頼できない接続

https://test-akito.kek.jp:9000/floating/ca/vac1.html

Google

**接続の安全性を確認できません**

test-akito.kek.jp:9000 に安全に接続するように求められましたが、接続の安全性が確認できませんでした。

安全に接続する場合は通常、あなたが適切な相手と通信することを確認できるように、信頼できる証明書を提供します。しかし、このサイトの証明書は信頼性を検証できません。

どうすればよいのか？

これまでこのサイトに問題なく接続できていた場合、このエラーが表示されるのは誰かがこのサイトになりすましている可能性があるということであり、接続すべきではありません。

[スタートページに戻る](#)

- ▶ [技術的詳細を表示](#)
- ▶ [危険性を理解した上で接続するには](#)

Google

日本

Google 検索 I'm Feeling Lucky

広告掲載 ビジネスソリューション プライバシーと利用規約 +Google Google について Google.com in English

UCHIYAMA Akito

+ 共有

Google

日本

Google 検索 I'm Feeling Lucky

広告掲載 ビジネスソリューション プライバシーと利用規約 +Google Google について Google.com in English

# RIBF 28GHz SC-ECRイオン源制御の例

28GECRIS Control GUI

vm1.ribf.local:9000/28gecris/

28GECRIS Control Main GAS Ext/Bias/Oven RF Bias/Oven PS Vacuum FC@R1

DATE: 12/07/29 09:31:56 Server Status: Connected.

GAS1 Control ( 02 )		VALUE	GAS2 Control ( 124Xe )		VALUE
OPEN	CLOSE	104.5	OPEN	CLOSE	104.3
Speed			Speed		
Speed Slow			Speed Slow		

Extraction Position		VALUE	Bias Disk		VALUE	Oven		VALUE
OUT	IN	36.7	OUT	IN	26.0	OUT	IN	99.9
Speed			Speed			Speed		
Speed Slow			Speed Slow			Speed Slow		

RF No.1		DAC	PF [W]	PR [W]	RF No.2 (Gyoltron)		DAC
DOWN	UP	0.0	0.1	-0.1	DOWN	UP	0.00
LV OFF					LV OFF		
HV OFF					HV OFF		

BIAS PS		DAC	Voltage [V]	Current [A]	Oven PS		DAC	Voltage
DOWN	UP	0.0	0.5	-0.008	DOWN	UP	0.0	-0.0

## Monitor

VAC Plasma	VAC Ext	VAC E11	VAC E21
4.4e-5 Pa	7.4e-3 Pa	1.2e-2 Pa	7.4e-5 Pa
Ext PS V [kV]	Ext PS A [mA]	Accel/Decel PS V [kV]	Accel/Decel PS A [mA]
0.0	0.0	0.0	0.0

28GECRIS Control GUI using Bootstrap - G...

vm1.ribf.local:9000/28gecris/fc.html

DATE: 12/07/29 09:31:56  
Server Status: Connected.



FC\_U10

10000

# クライアントサイド実装(モバイル)



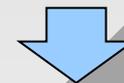
Safari on iPhone4S



Google Chrome on Android4.0

Objective-C + cocoa Touchより  
難易度は大幅に下がる

・ SafariにはiPhone/iPod Touch向けに  
タッチイベントを取得する関数がある



マルチタッチアプリケーションの開発も可能

# まとめ

- ✓ 新しいHTMLの規格であるWebSocketからEPICS Channel Accessを行う事が可能になった
- ✓ 実現する為に
  - Node.jsからCAを利用するためにNodeCAを開発した
  - クライアントサイドからのCA要求に対応するWebSocketサーバを開発した
- ✓ ブラウザから制御を行い従来のアプリケーションと同程度のインタラクティブ性を実現した
- ✓ PCでなくiPhone4SやAndroidのブラウザからもWebSocketで制御できる事を確認した

---

ご清聴ありがとうございました