EWD.js

Rob Tweed
M/Gateway Developments Ltd

Twitter: @rtweed
Email: rtweed@mgateway.com

EWD.js

- Background, History and Aims
- Underlying Technology & Architecture
  - Comparison with “classic” EWD
  - Benefits
- How do you use it?
  - Examples

The backdrop

- How to make Mumps acceptable to the “mainstream”?
- How to recruit and retain new developers with the expertise to:
  - exploit the new and expanding possibilities of browsers as the modern, ubiquitous UI platform
  - make VistA the healthcare Big Data visualisation platform of choice
  - make Healthcare IT as exciting and cool as the gaming and social media industries?

http://robtweed.wordpress.com

THE EWD FILES

ABOUT

Enterprise Web Development: Javascript, NoSQL and Big Data

CAN A PHOENIX ARISE FROM THE ASHES OF MUMPS?

January 22, 2013 · *by* robtweed · *in Uncategorized*

There's a major problem that is growing increasingly critical in the Mumps application world: where are the new generation of developers going to come from to support what is a pretty massive legacy of applications? The US Dept of Veterans' Affairs' (VA) *VistA* Electronic Healthcare Record (EHR) is just one of a large number of healthcare applications that was written in Mumps and is supported and maintained by a dwindling number of developers who understand and/or enjoy using Mumps. The sheer scale of VistA and its growing popularity as an Open Source EHR outside the VA means severe problems ahead if a way of

Search this site...

RECENT POSTS

- EWD Course: 17-19 May, Fairfax VA
- Mumps: the proto-database (or how to build your own NoSQL database)
- InterSystems Global Summit: Come and See the Show!
- The Uncertainty Principle
- Writing EWD Applications entirely

M/GATEWAY DEVELOPMENTS

Mumps: a Dead End?

- Not seen as a sensible career move by most of the new generation of developers
  - Unlikely to want a Mumps development job
  - Unlikely to be retained very long
- Dwindling pool of Mumps skills, despite training efforts
- Significant brake on VistA uptake:
  - Seen as a risk by potential adopters:
    - Old, obsolete technology
    - Shortage of developers

Understanding the bigger picture

- Google Alerts
- Twitter searches & dialogue
- Feedback on articles and comments
- Feedback from other communities

- Understanding the issues from the point of view of those looking at Mumps from the mainstream

M/GATEWAY DEVELOPMENTS

For example


teemuvesala
#MUMPS worse than #Perl
github.com/OSEHRA/VistA-M…

For example

file | 82 lines (81 sloc) | 5.392 kb

Edit | Raw | Blame | History

```
1  OOPSPC41 ;HIRMFO/YH-EMPLOYEE DATA, CA2 FORM ;6/14/98
2   ;;2.0;ASISTS;;Jun 03, 2002
3   ;EMPLOYEE DATA
4   ;EMPLOYEE'S DATA
5   S OOPSDATA=$P($G(^OOPS(2260,IEN,0)),"^",2)
6   W !,"PU0.8,25.5;LB"_OOPSDATA_"@;" ;NAME
7   S OOPSDATA=$G(^OOPS(2260,IEN,"2162A"))
8   S OOPSP=$P(OOPSDATA,"^") I OOPSP'["-" S OOPSP=$E(OOPSP,1,3)_"-"_$E(OOPSP,4,5)_"-"_$E(OOPSP,6,13)
9   W !,"PU15.9,25.5;LB"_OOPSP_"@;" ;SSN
10  S OOPSP=$P(OOPSDATA,"^",2) I OOPSP'="" D WDATE^OOPSPUT1(OOPSP,"3.1,24.8","3.9,24.8","4.8,24.8") ;DATE OF BIRTH
11  S OOPSP=$P(OOPSDATA,"^",3) W !,"PU6.5,24.7;LB"_$S(OOPSP=1:"Male",OOPSP=2:"Female",1:" ")_"@;" ;SEX
12  N PHN
13  S PHN=$TR($P(OOPSDATA,"^",8),"/-*#","")
14  W !,"PU8.7,24.7;LB"_$E(PHN,1,3)_"-"_$E(PHN,4,6)_"-",$E(PHN,7,10)_"@;"
15  W !,"PU16.6,24.7;LB"_$S($P(OOPSDATA,"^",12)'="":+$P(OOPSDATA,"^",12),1:"")_"@;PU18.7,24.7;LB"_$P(OOPSDATA,"^",1
16  W !,"PU0.8,23.5;LB"_$P(OOPSDATA,"^",4)_"@;PU0.8,22.7;LB"_$P(OOPSDATA,"^",5)_", "_$S($D(^DIC(5,+$P(OOPSDATA,"^",
17  W !,"PU12.8,22.7;LB"_$P(OOPSDATA,"^",7)_"@;" ;ADDRESS
18  S OOPSP=+$P($G(^OOPS(2260,IEN,"CA2A")),"^",8) ;DEPENDENTS
19  I OOPSP>0,OOPSP<6 W !,$S(OOPSP=1:"PU16.2,23.8;LBX@;",OOPSP=2:"PU16.2,23.4;LBX@;",OOPSP=3:"PU16.2,23;LBX@;",OOPS
20  I OOPSP>5,OOPSP<8 W !,$S(OOPSP=6:"PU16.2,23.4;LBX@;PU16.2,23;LBX@;",1:"PU16.2,23.8;LBX@;PU16.2,23.4;LBX@;PU16.2
21  W !,"PU0.8,20.9;LB"_$P($G(^OOPS(2260,IEN,"CA2A")),"^",9)_"@;"
22  ; Patch 8
23  S OCC=$$GET1^DIQ(2260,IEN,15,"E")
24  S OCC=$S(OCC<2200:"G"_OCC,(OCC>2499&(OCC<9001)):"W"_OCC,(OCC=9999):"Z"_OCC,1:"")
25  W "PU16.2,21.1;LB"_OCC_"@;" K OCC        ; OCCUPATION CODE
```

M/GATEWAY DEVELOPMENTS

VistA: Why it's like it is

- When first written, Mumps had many constraints:
  - 8 character variable name
  - 8 character global names
  - 8 character routine names
  - 8 character labels
  - No variable scoping
  - No functions
  - Everything upper case

VistA: Why it's like it is

- Hardware limitations at the time:
  - Very expensive
  - Very low-powered hardware
  - Very little memory
  - Developer time relatively less costly
- Code written to:
  - minimise resource usage
  - Maximise performance
  - At the expense of maintainability

Write code that doesn't suck?

- Modern Mumps coding has none of the original limitations
- Well-written modern Mumps code can be highly readable, understandable and maintainable

But:

- VistA coding standards: SAC Compliance
  - Retains most of the coding limitations unnecessarily
- Original VistA Mumps code depended on leaky, globally-scoped variables
  - Very difficult to build new, properly-scoped code to work with leaky legacy code
  - Huge task to rewrite properly

The poisonous conflation

- Mumps is positioned as a language
- It's also a database
- Mention Mumps to the mainstream and they focus on the language, not the database
- Google search for Mumps language and what do you find almost immediately?

The Poison


THE DAILY WTF
Curious Perversions in Information Technology
Sign On • Join • Forums
Google™ Custom Search
Search
Home » Articles » Feature Articles » A Case of the MUMPS
New Relic.
It was the #Feed action
In the index.php
With the ...Empty delimiter
Stop guessing, we'll show you
Get Started Now
SUBMIT YOUR WTF
Content
Random Article
All Articles
Feature Articles
CodeSOD
Error'd
Tales from the Interview
Alex's Soapbox
Free WTF Swag!
« Local Mistakes
One Step Forward... »
A Case of the MUMPS 2007-02-13
by Alex Papadimoulis in Feature Articles (357 Comments)
You may not realize it, but the majority of us developers have been living a sheltered professional life. Sure, we've got that living disaster of a C++ application and that ridiculous interface between PHP and COBOL written by the boss, but I can assure you, that all pales in comparison to what many, less fortunate programmers have to work with each day. These programmers remain mostly forgotten, toiling away at a dead-end career maintaining ancient information systems whose ridiculously shoddy architecture is surpassed only by the tools used to create it. Bryan H lived in such a world for over two years. Specifically, he worked at a "MUMPS shop."

Back comes the poison

Andrew Clegg @andrew_clegg 27 Mar

I think this is why no-one uses it ow.ly/jsVcb ... RT @al3xandru: Mumps: The Proto-Database nosql.mypopescu.com/post/464079714

M/GATEWAY DEVELOPMENTS

The reason for the poison?

- Written by guys who have had to maintain the leaky code in Mumps applications such as Epic and VistA
- They believe (and perpetuate the belief) that it's how you have to code in the Mumps language
  - nobody told them why that code was the way it was, and that it didn't need to be that way
  - now all modern developers believe it too!

Then we have the Misconceptions

Alan C. Viars @aviars 19 Feb

@rtweed @jeffbrandt I get the comparison and understand it widely used in legacy systems, but I wouldn't start a new project on UNIVAC

Expand Reply Retweet Favorite ••• More

Alan C. Viars @aviars 19 Feb

@rtweed @jeffbrandt MongoDB is not new and shinny at this point. Very proven. Interfaces in all langs.

Expand

Alan C. Viars @aviars 19 Feb

@rtweed @jeffbrandt MUMPS problems: Lack of interfaces for most langs. No support for XML/JSON. Built before TCP/IP. Small community.

View conversation

More misconceptions

jeffbrandt @jeffbrandt 18 Feb

RT @rtweed: @aviars Why not use cobol? it works but.... limiting How many mumps programmer? How many schools teach MUMPS, Madison

Expand

jeffbrandt @jeffbrandt 18 Feb

RT @rtweed: @aviars I'd be interested to hear your reasoning behind that conclusion 1960 technology, Silo data, no interfaces...

Expand

jeffbrandt @jeffbrandt 18 Feb

RT @aviars: Mumps: <- Anyone building something new, MUMPS over MongoDB needs head... | you can drive your car w/ your feet but Why :)

Expand

jeffbrandt @jeffbrandt 18 Feb

RT @aviars: RT @Mumps Univ NoSQL DB <- Anyone building something new on MUMPS over MongoDB needs their head examined. |Agree 100%

Expand

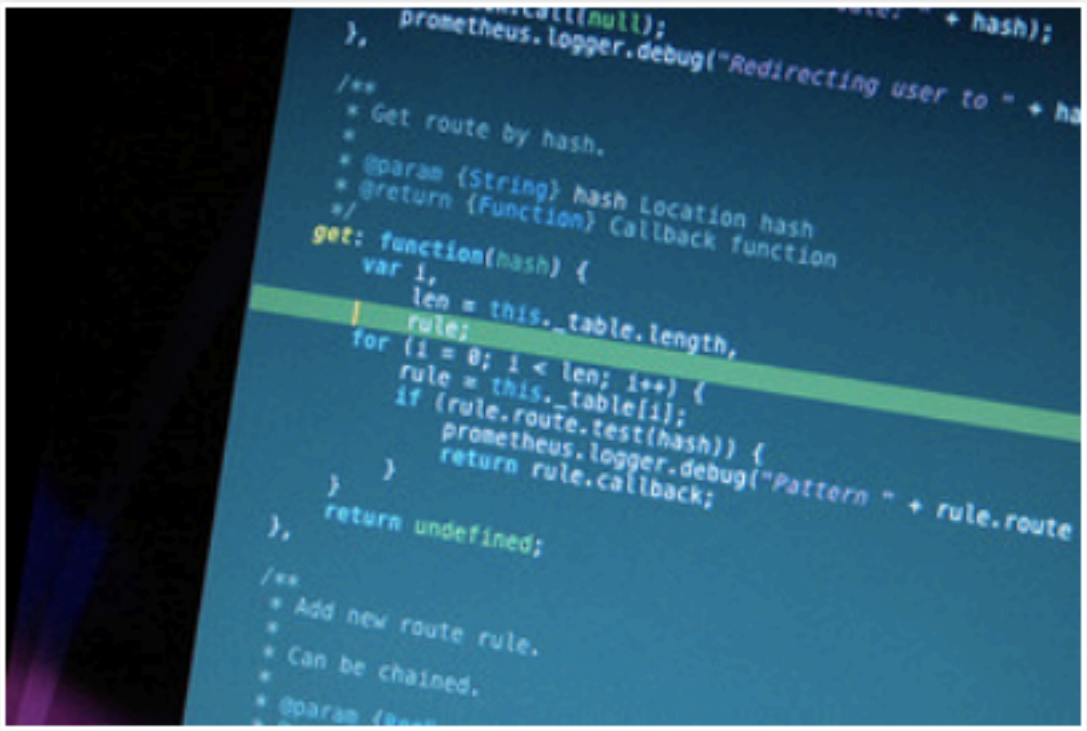
M/GATEWAY DEVELOPMENTS

Reason

- Epic
  - I don't know if criticisms of Epic's interoperability are founded or not, but if they are, it's a commercial decision by Epic, not a technical limitation of Mumps
  - However, this is naively being translated into a poisonous "truth" that Mumps has significant technical limitations
- Now being turned into a broader "truth":
  - Mumps' dominance in the sector is the root of all ills in Healthcare IT
    - ie it's old, out-of-dated technology that is fundamentally flawed and needs replacing

www.hitconsultant.net/2013/04/30/is-mumps-infecting-ehrs/

Is MUMPS Infecting EHRs?



CareCloud's Ahmed Mori questions the effectiveness of the MUMPS EHR programming language in healthcare.

The word 'mumps' has two meanings in the healthcare industry. Most popularly, mumps is a virus characterized by a painful swelling of the salivary glands, and is rather contagious.

Its other usage is an acronym for Massachusetts General Hospital Utility Multi-Programming System, a programming language that is **almost**

Step 1: remove the conflation

- Focus on Mumps, the database
  - the Mumps language is pilloried by the mainstream, for founded and unfounded reasons
    - attempting to change that opinion is futile
  - By Mumps being considered a language, the Mumps database is dismissed by the mainstream
  - It's the database that is the unique and powerful part of the technology

Mumps: Universal NoSQL

www.mgateway.com/docs/universalNoSQL.pdf

A Universal NoSQL Engine, Using a Tried and Tested Technology

Rob Tweed (*rtweed@mgateway.com* **web:** *http://www.mgateway.com*)
George James (*GeorgeJ@georgejames.com* **web:** *http://www.georgejames.com*)

Introduction

You wouldn't expect a programming language from the 1960s to have anything new to teach us, especially one that diverged from the mainstream around the time that Dartmouth BASIC became popular. Even more especially a programming language called MUMPS.

However, surprisingly there is one aspect of this archaic language that is still ahead of it's time. MUMPS has a pearl in its oyster called Global Persistent Variables. These are an abstraction of the B-tree structures that are normally used by MUMPS to store large volumes of data. Global Persistent Variables (usually simply referred to as "Globals") are an expressive and highly efficient way of modelling all of the common use cases that are targeted these days by NoSQL databases.

M/GATEWAY DEVELOPMENTS

Step 2: Replace the language

- A language that has similar good parts to Mumps
- Object oriented, but dynamic objects
- High performance and scalability
- Highly popular language that is likely to stay that way
- Similar intimate integration with the Mumps database

JavaScript: ticks all the boxes

- A language that has similar good parts to Mumps
- Object oriented, but dynamic objects
- High performance and scalability
- Highly popular language
  - Games, 3-d, social media industries
  - It's not going to disappear
  - It's the language new developers are learning

JavaScript runs in the browser, right?

- Yes, but now also on the server:
  - Node.js

Node.js

- Server-side implementation of Javascript
- Started as an open-source project by Ryan Dhal
- Uses Google's V8 Javascript engine
- Designed for network programming
  - Event-driven
  - Normally asynchronous I/O
- Very high performance and scalability
- Runs on Linux, Windows & Mac OS X
- Recently officially approved for use by the VA

Node.js

- Many open-source modules, in particular:
  - built-in Web server
  - Socket.io: web-sockets

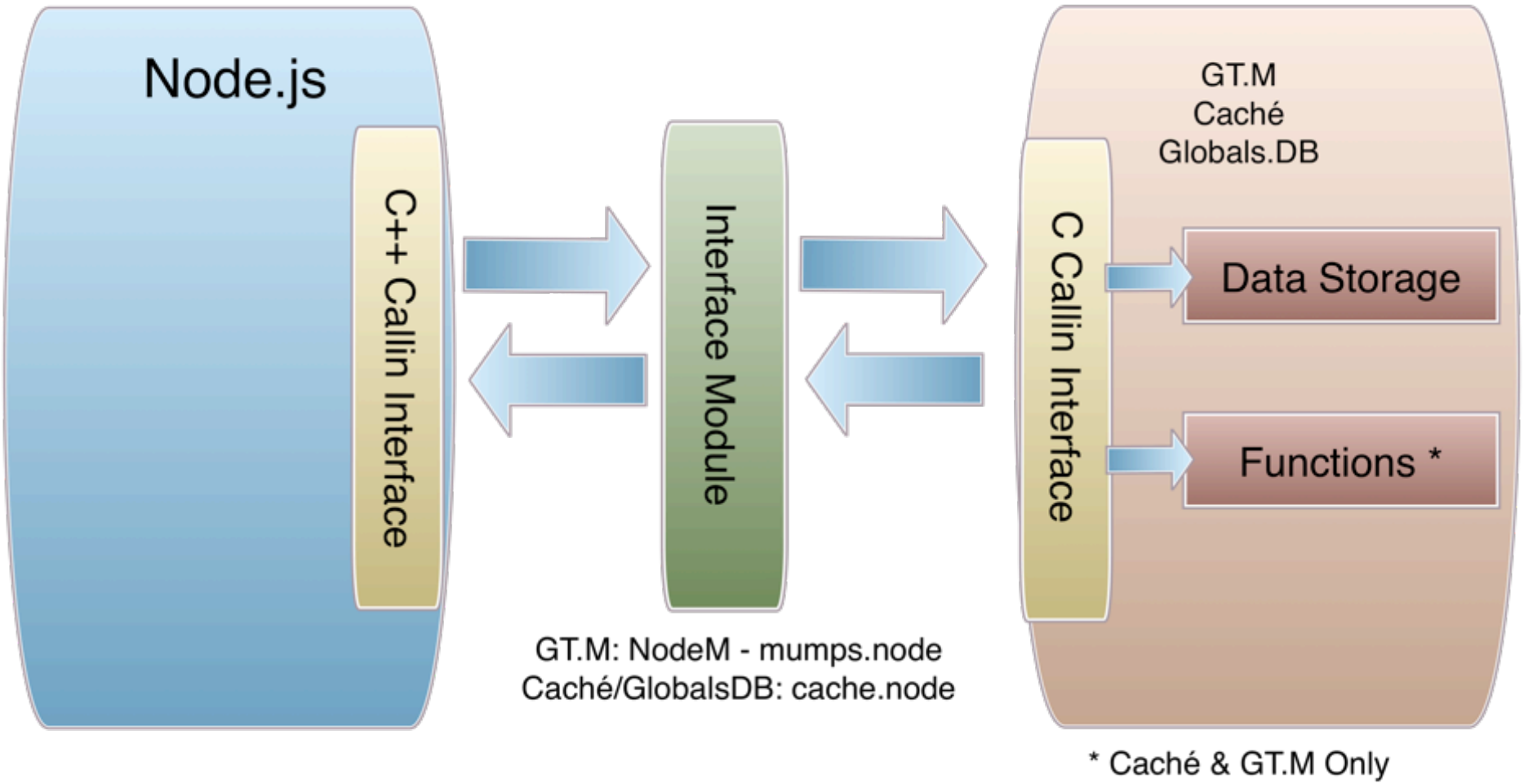
InterSystems Node.js Interface

- Initially introduced for their free Globals database
  - Used their very low-level interface into the core global database engine
  - Written by Chris Munt (M/Gateway Developments)
  - Very high performance
- Ported to Caché (2012.2)
  - Official standard interface
  - function() API added to invoke Mumps functions from Javascript

Equivalent for GT.M?

- David Wicksell: NodeM
  - https://github.com/dlwicksell/nodem
  - API-compatible reverse-engineer of InterSystems' Node.js interface
    - Behaves identically

Node.js Interface



Mumps, the JSON Database

- Important to present a Mumps database as something natural to a JavaScript developer
- JSON is their lingua-franca
- Abstract Mumps Global storage into persistent JavaScript Objects
  - similar level of “intimacy” as between the Mumps language and database

Dynamic OO Global Abstraction

- Underlying concept: any object and its properties can be easily modelled as a corresponding Mumps Global node, eg:
  - patient.address.town
  - patient.address.zip
  - patient.name
  - ^patient(id,”address”,”town”)=“New York”
  - ^patient(id,”address”,”zip”)=123456
  - ^patient(id,”name”)=“John Smith”

Document Storage

- Physical Global Nodes projected as *GlobalNode* objects in JavaScript
- *GlobalNode* objects have two key methods:
  - *_setDocument()*
    - Maps a JSON document into an equivalent global as a sub-tree
  - *_getDocument()*
    - Maps a GlobalNode's sub-tree into a JSON object

JavaScript Document Storage

```
var gridData = [
   {col1: 1, col2: 1, name: ‘rec1’},
   {col1: 4, col2: 4, name: ‘rec4’}
 ];
 session.$('newGridData')._setDocument(gridData);
```

JavaScript Document Storage

```
var gridData = [
  {col1: 1, col2: 1, name: ‘rec1’},
  {col1: 4, col2: 4, name: ‘rec4’}
];
session.$('newGridData')._setDocument(gridData);
```

```
^%zewdSession("session",4020,"newGridData",0,"col1")=1
^%zewdSession("session",4020,"newGridData",0,"col2")=1
^%zewdSession("session",4020,"newGridData",0,"name")="rec1"
^%zewdSession("session",4020,"newGridData",1,"col1")=4
^%zewdSession("session",4020,"newGridData",1,"col2")=4
^%zewdSession("session",4020,"newGridData",1,"name")="rec4"
```

JavaScript Document Storage

```
^%zewdSession("session",4020,"newGridData",0,"col1")=1
^%zewdSession("session",4020,"newGridData",0,"col2")=1
^%zewdSession("session",4020,"newGridData",0,"name")="rec1"
^%zewdSession("session",4020,"newGridData",1,"col1")=4
^%zewdSession("session",4020,"newGridData",1,"col2")=4
^%zewdSession("session",4020,"newGridData",1,"name")="rec4"
```

```
var gridData = session.newGridData._getDocument(gridData);

[
   {col1: 1, col2: 1, name: 'rec1'},
   {col1: 4, col2: 4, name: 'rec4'}
 ];
```

The key ingredients

- Mumps as one of the most powerful and proven NoSQL databases available:
  - high scalability, high performance
- Projected as a JSON database
- Accessible via JavaScript
  - using Node.js to provide server-side environment
  - using native interface for Cache
  - or equivalent interface for GT.M

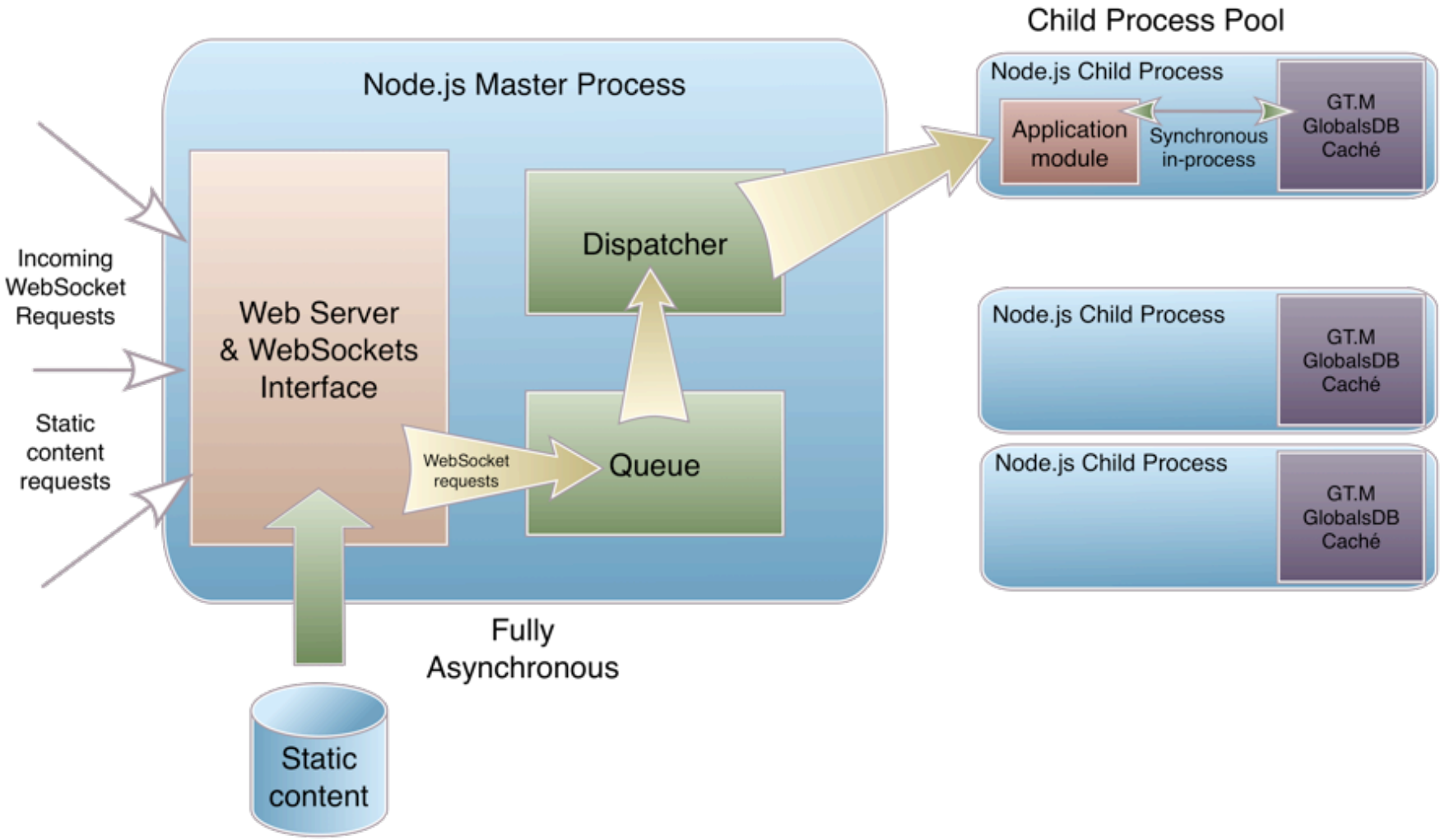
One more key piece of magic

- WebSockets
  - HTML5 standard
  - bi-directional, event-driven socket connection
    - between browser & back-end
  - JSON messaging between browser and back-end
  - generally being found to be faster than HTTP/ Ajax
  - No polling!

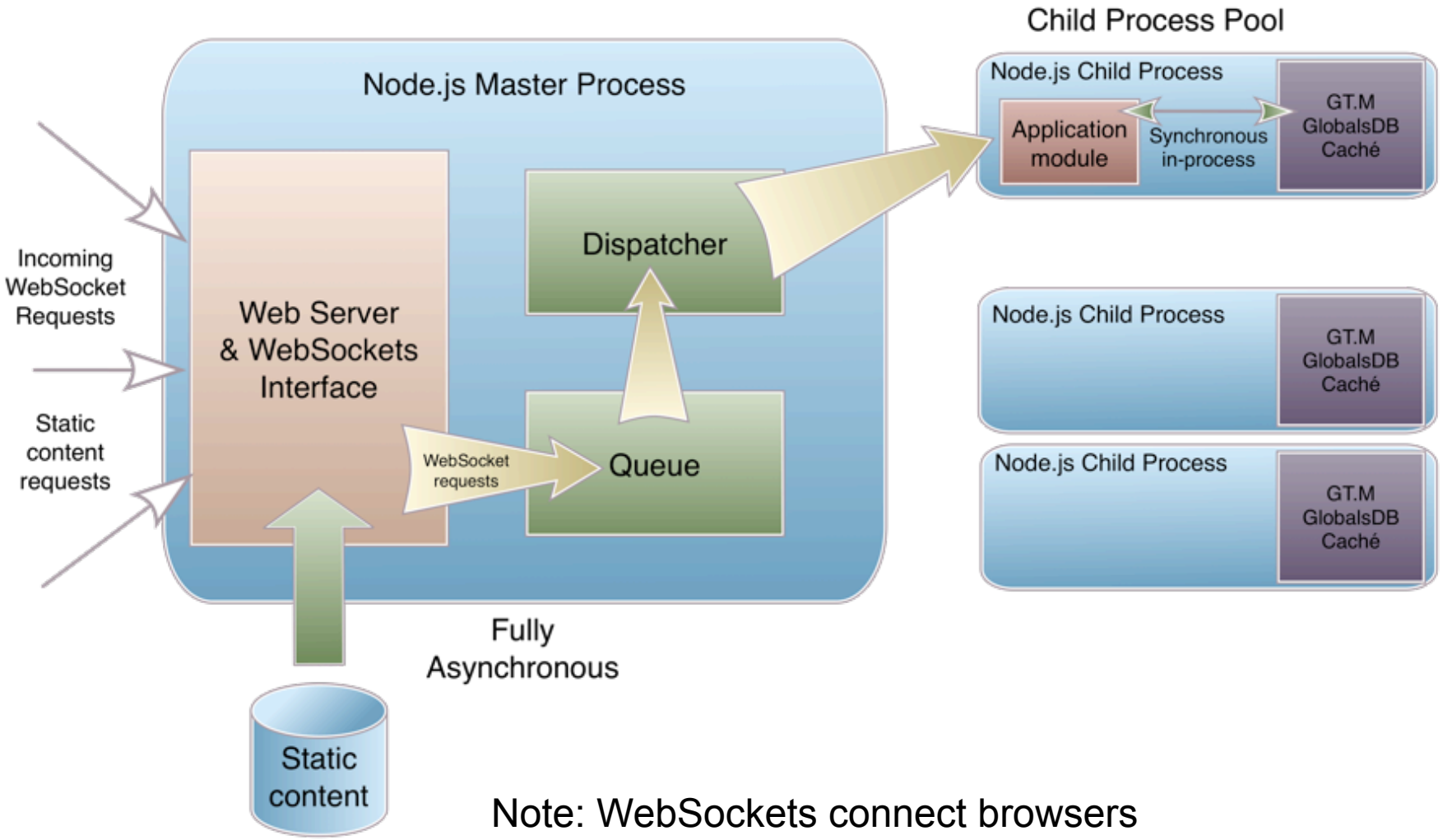
Package it all up

- EWD.js
  - framework for JavaScript applications
  - 100% JavaScript & JSON
  - WebSockets to deliver JSON between browser and Node.js back-end
  - Mumps databases abstracted to appear to be JSON stores
  - fully event-driven
  - client/server in the browser

EWD.js Architecture

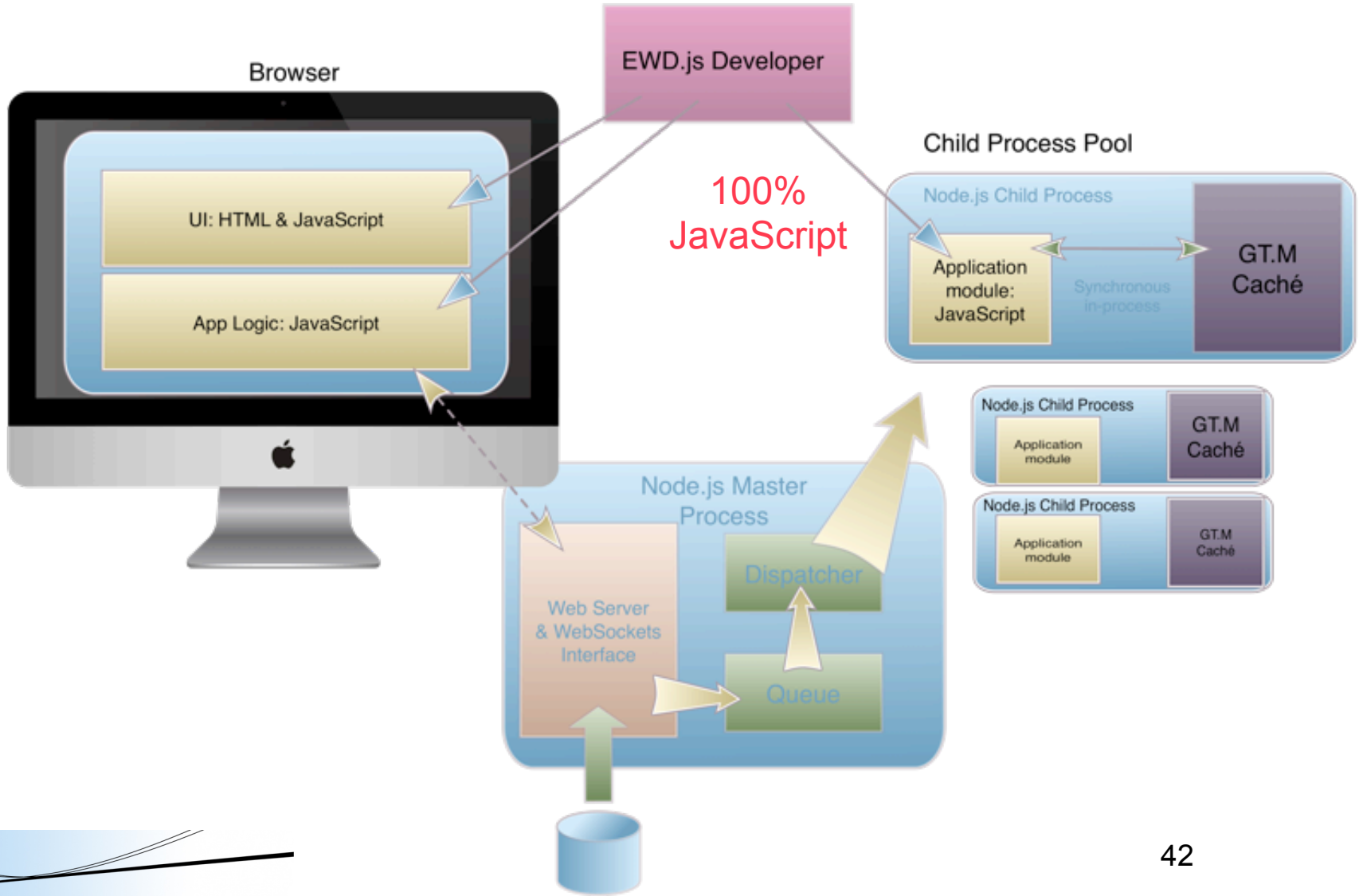


EWD.js Architecture



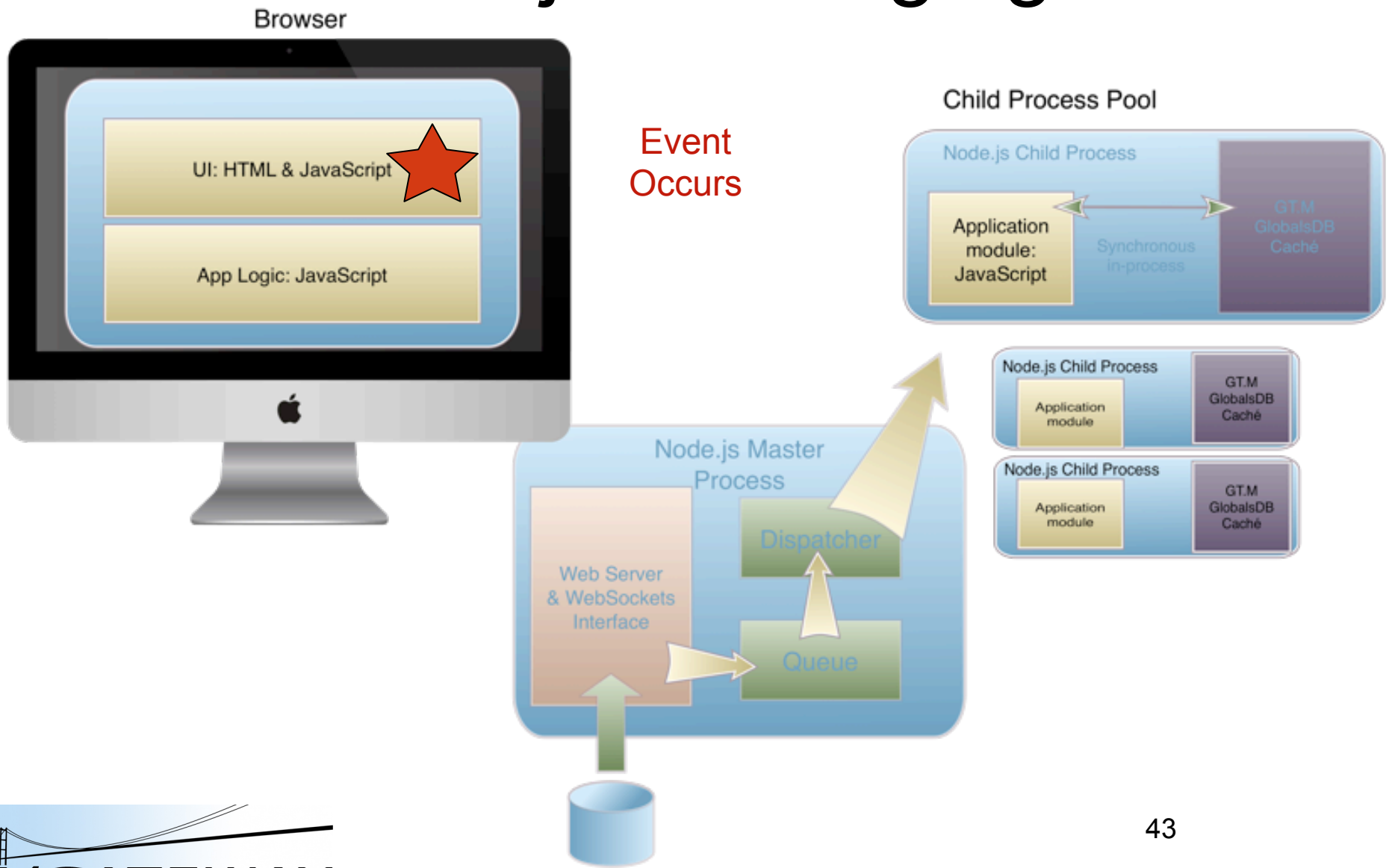
Note: WebSockets connect browsers to Node.js Master Process, not to Caché or GT.M

EWD.js Development

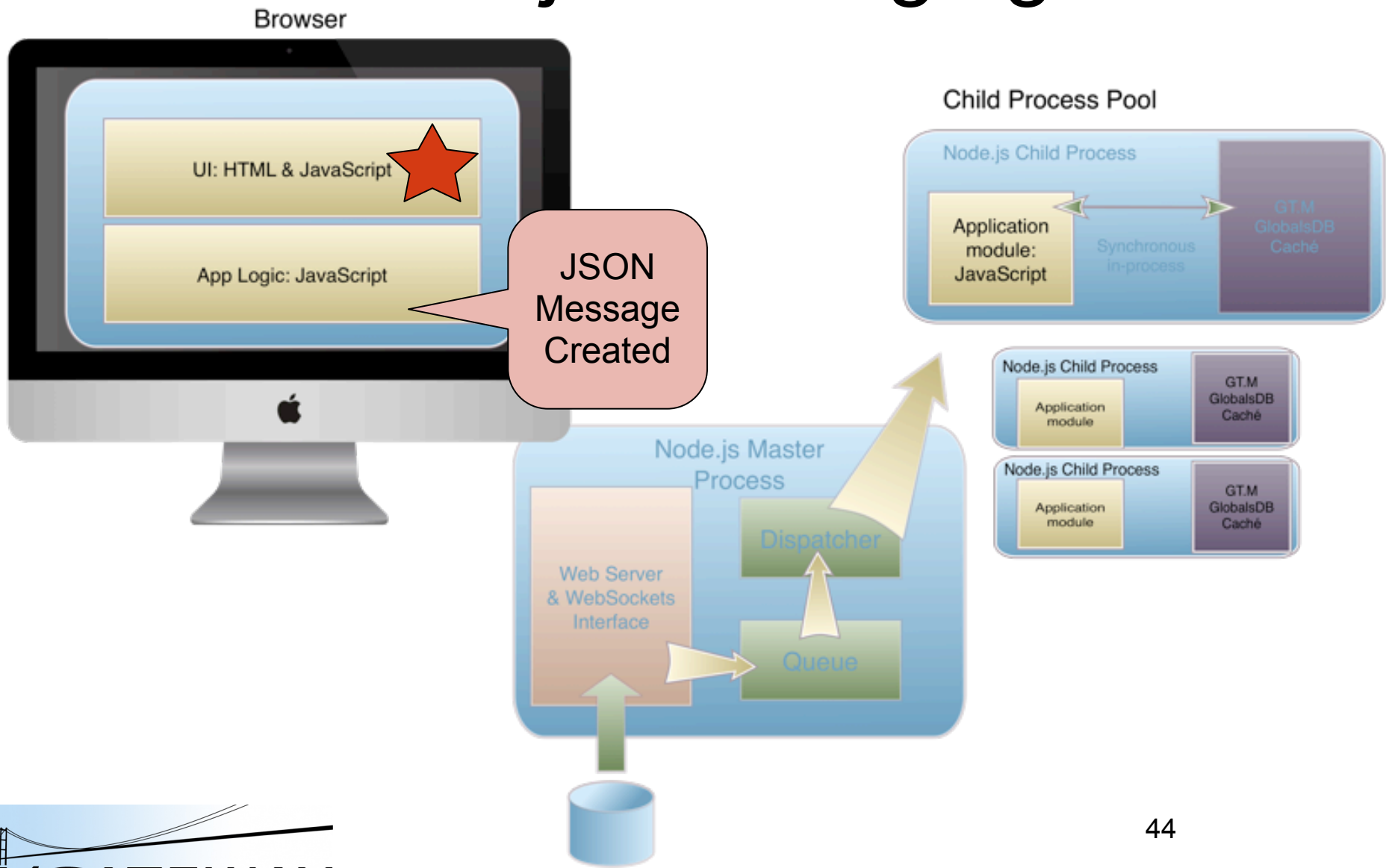


M/GATEWAY DEVELOPMENTS

EWD.js Messaging

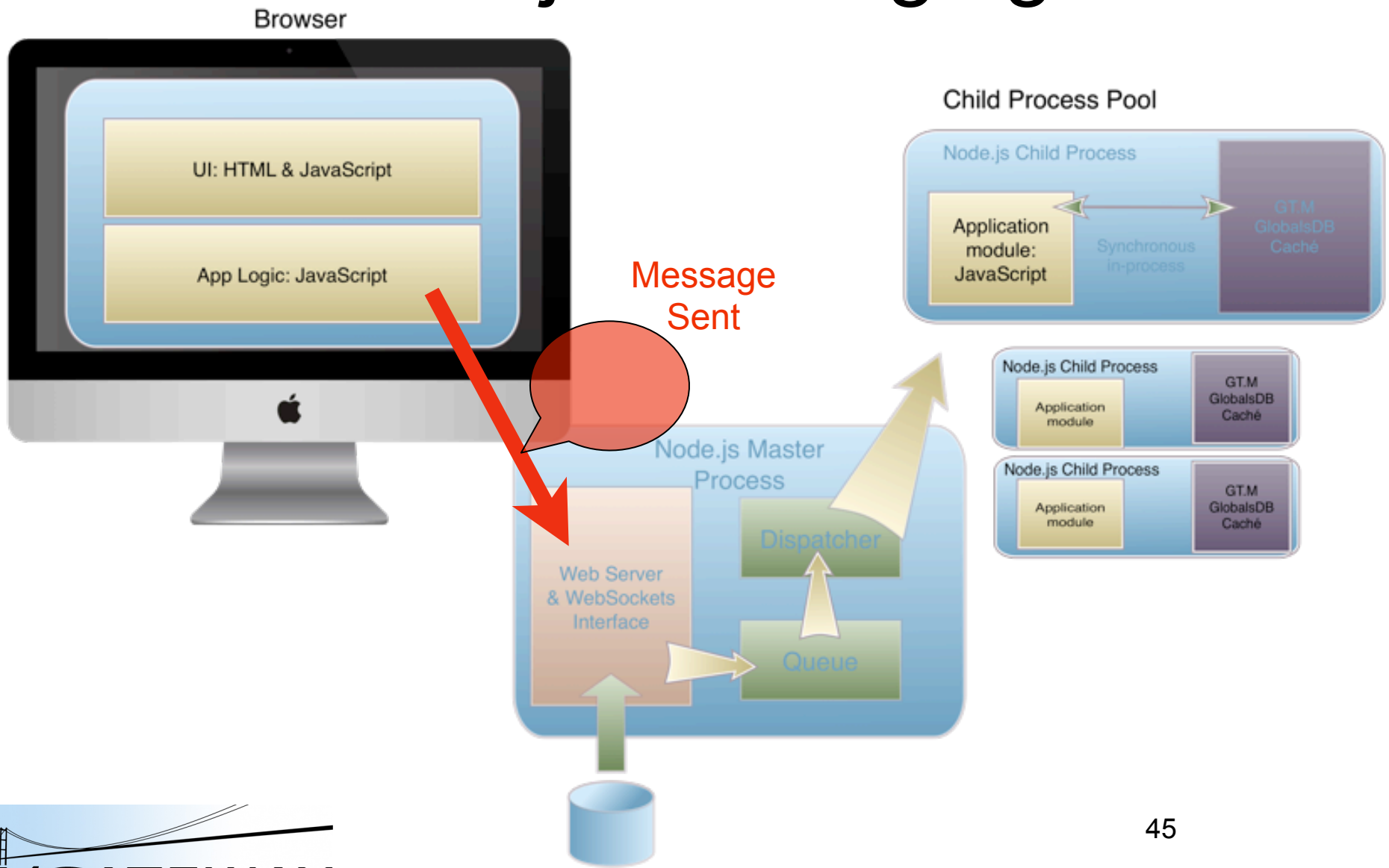


EWD.js Messaging

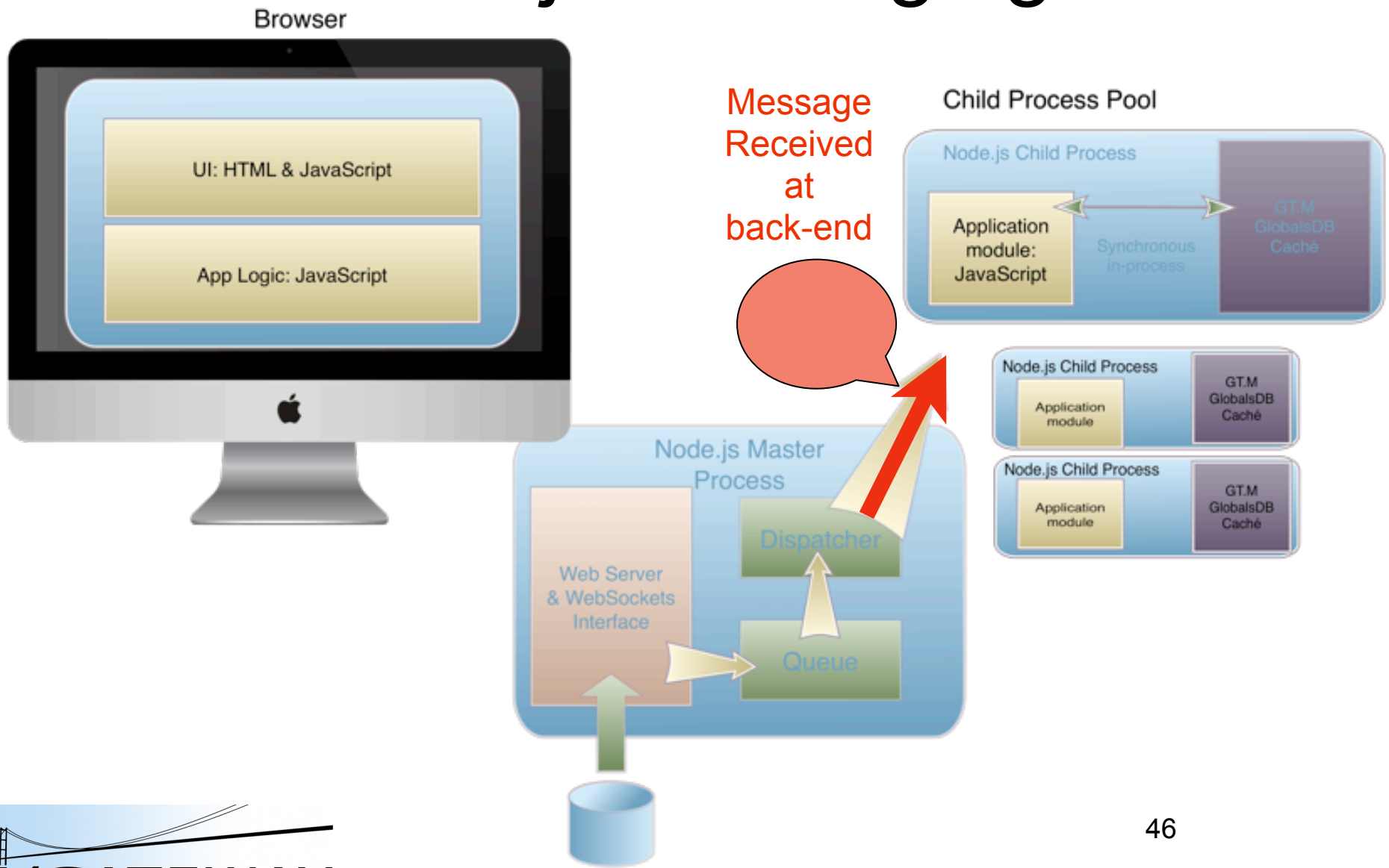


M/GATEWAY DEVELOPMENTS

EWD.js Messaging

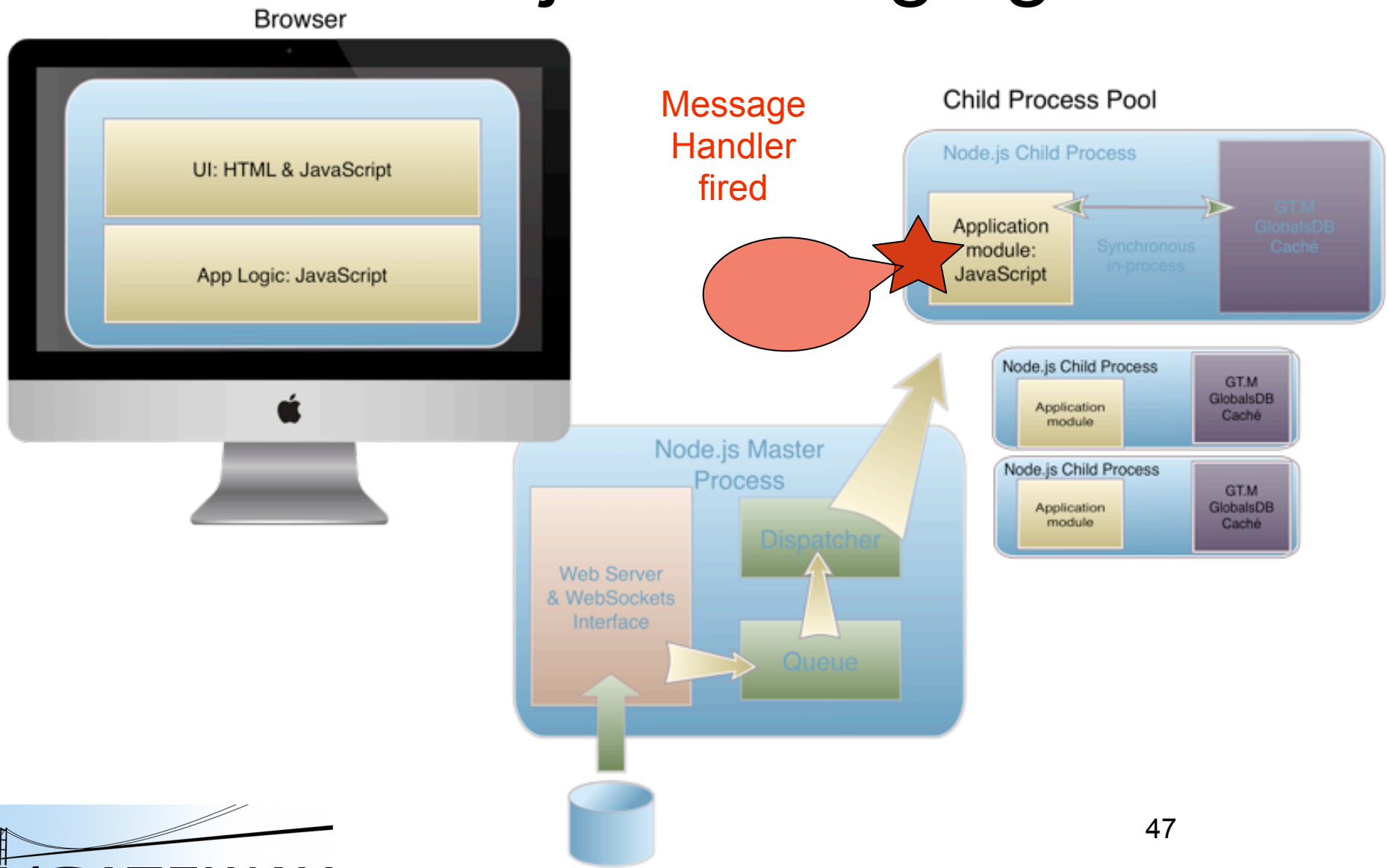


EWD.js Messaging



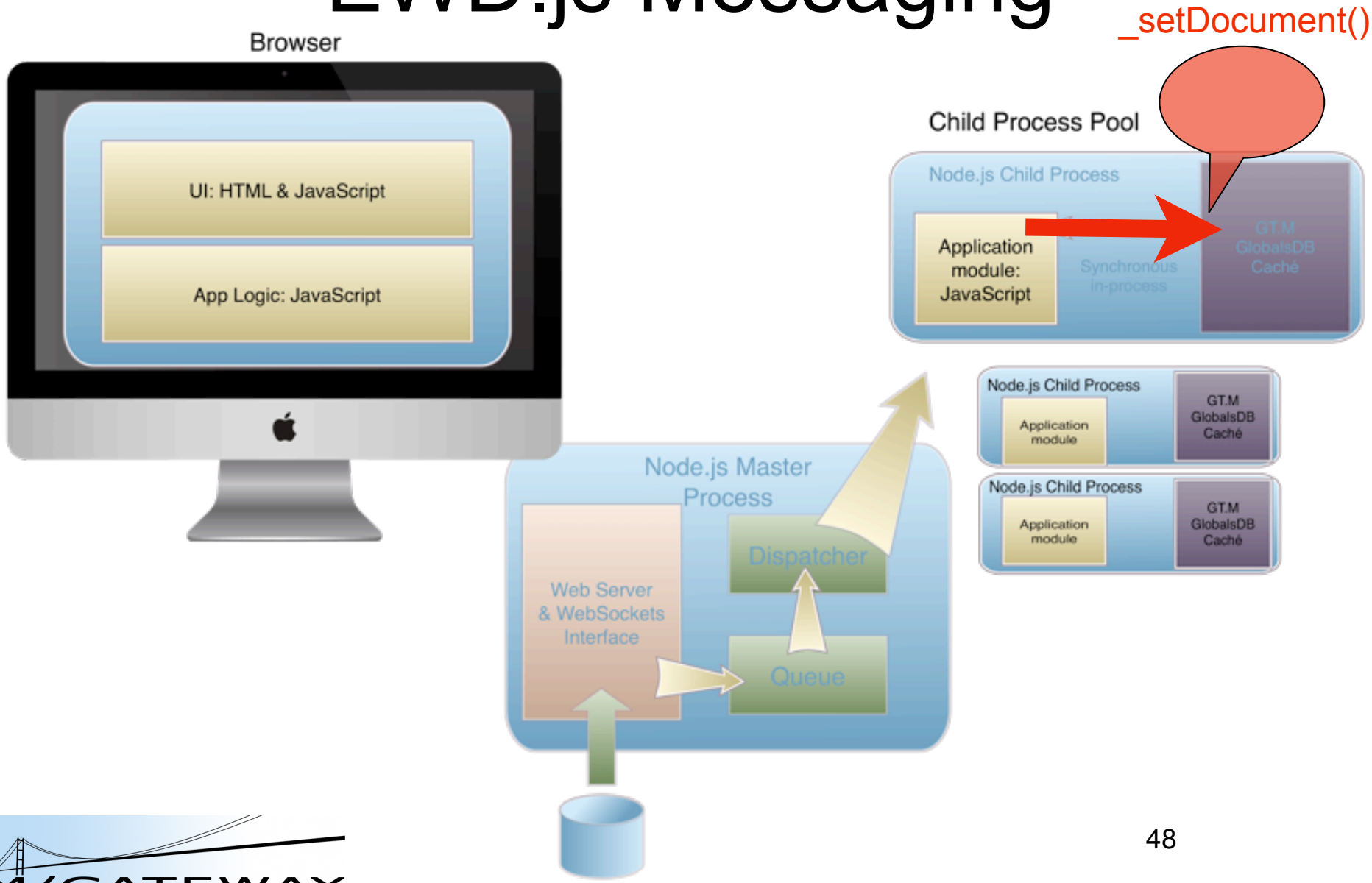
M/GATEWAY DEVELOPMENTS

EWD.js Messaging



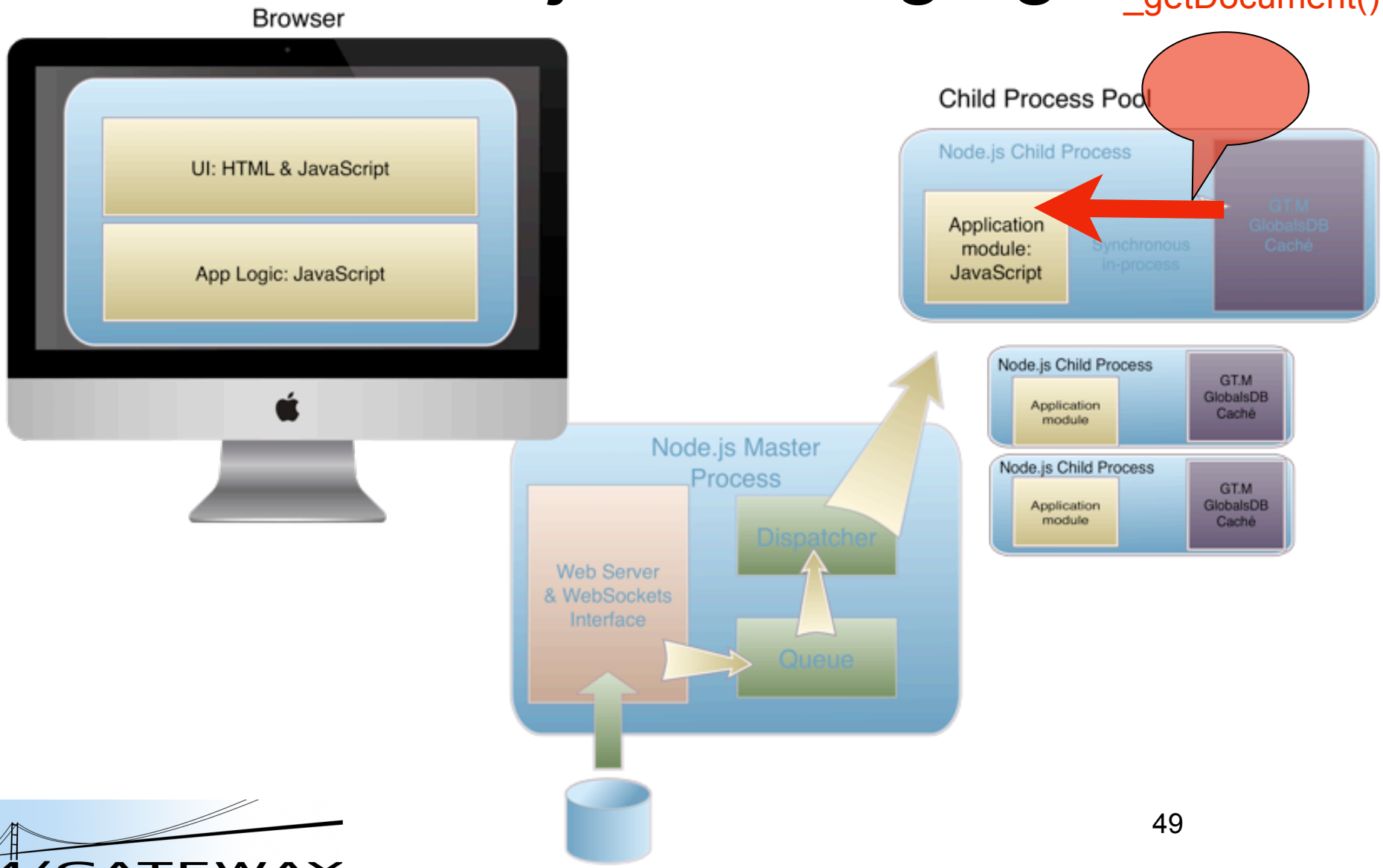
M/GATEWAY DEVELOPMENTS

EWD.js Messaging



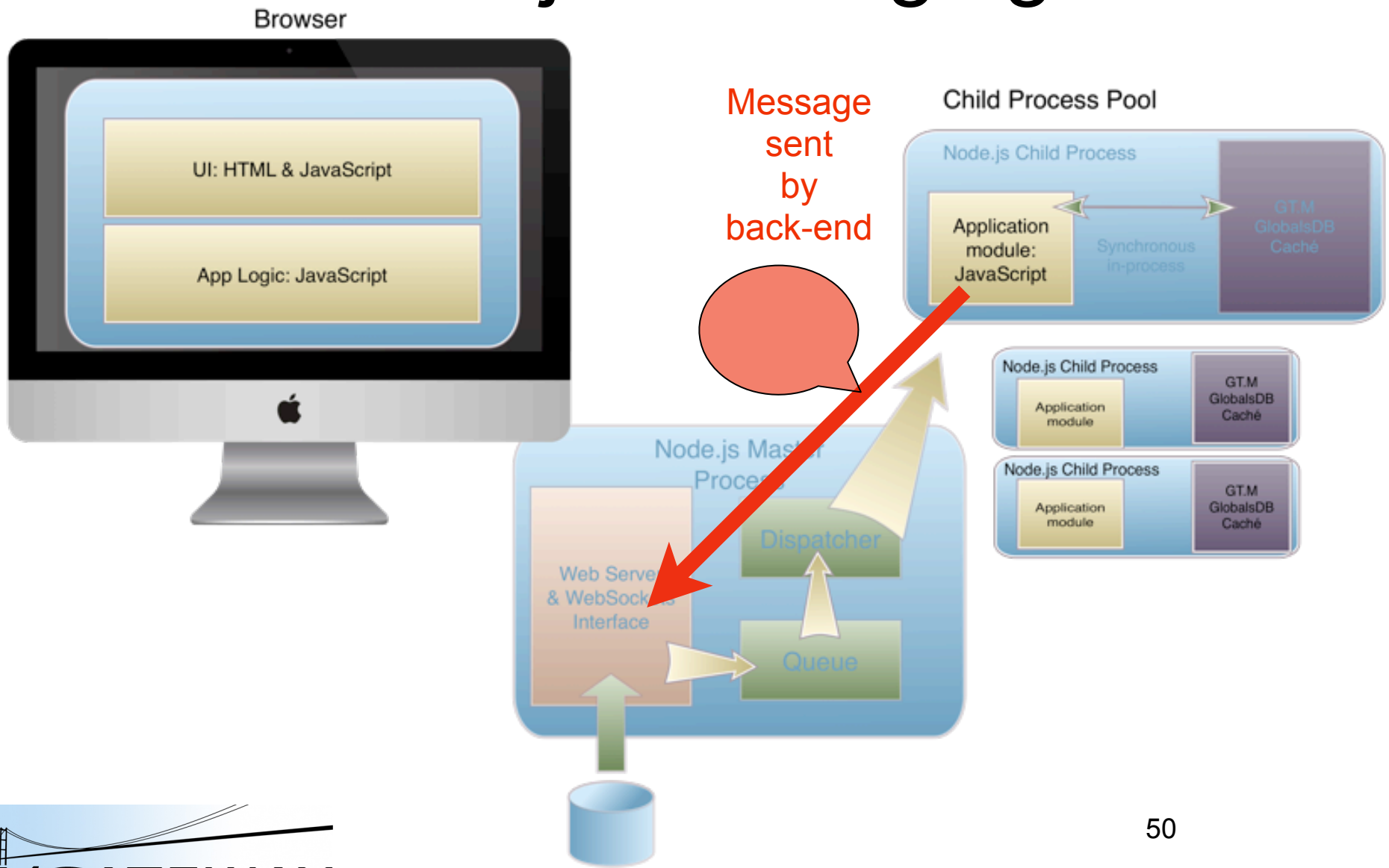
M/GATEWAY DEVELOPMENTS

EWD.js Messaging



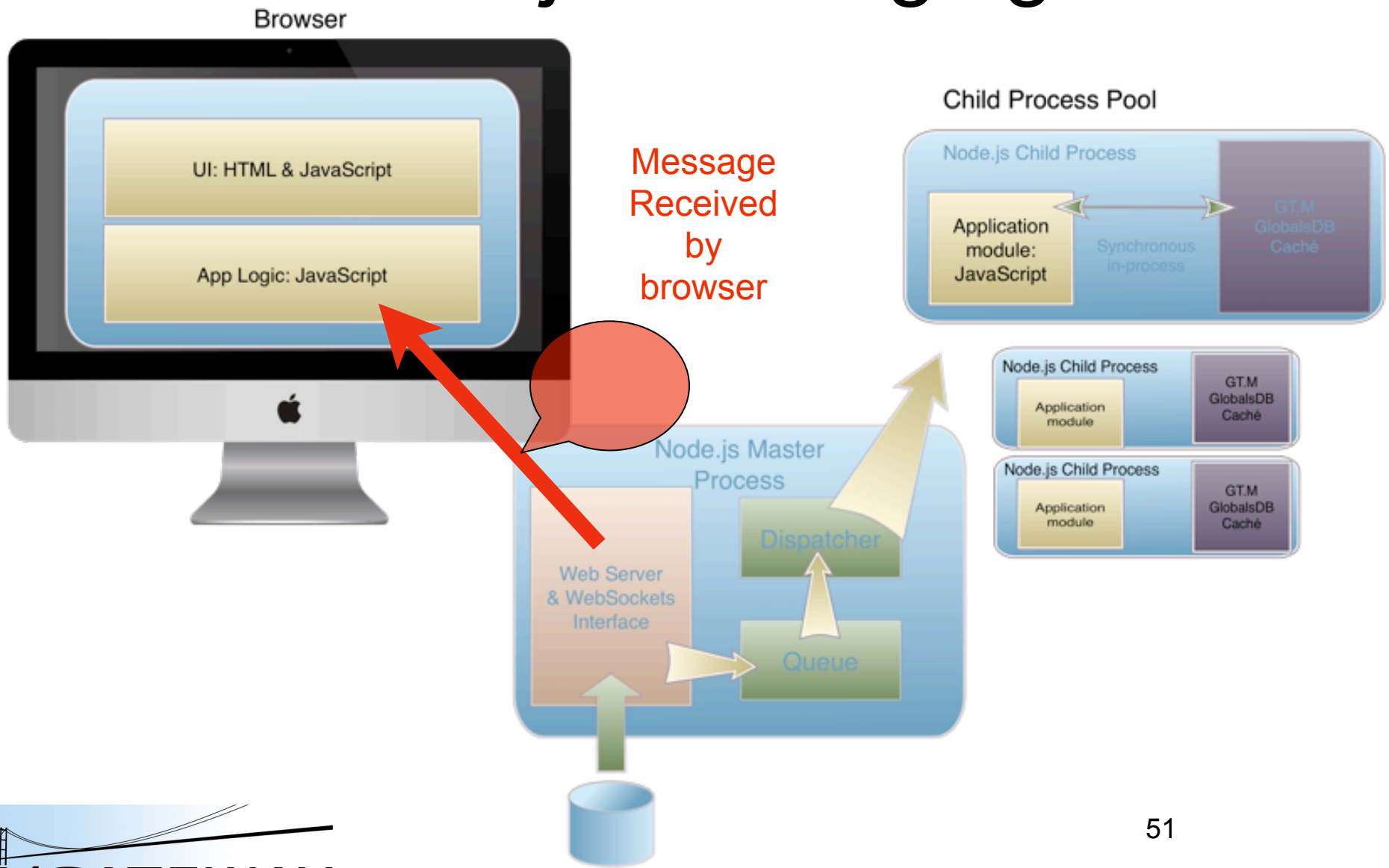
M/GATEWAY DEVELOPMENTS

EWD.js Messaging

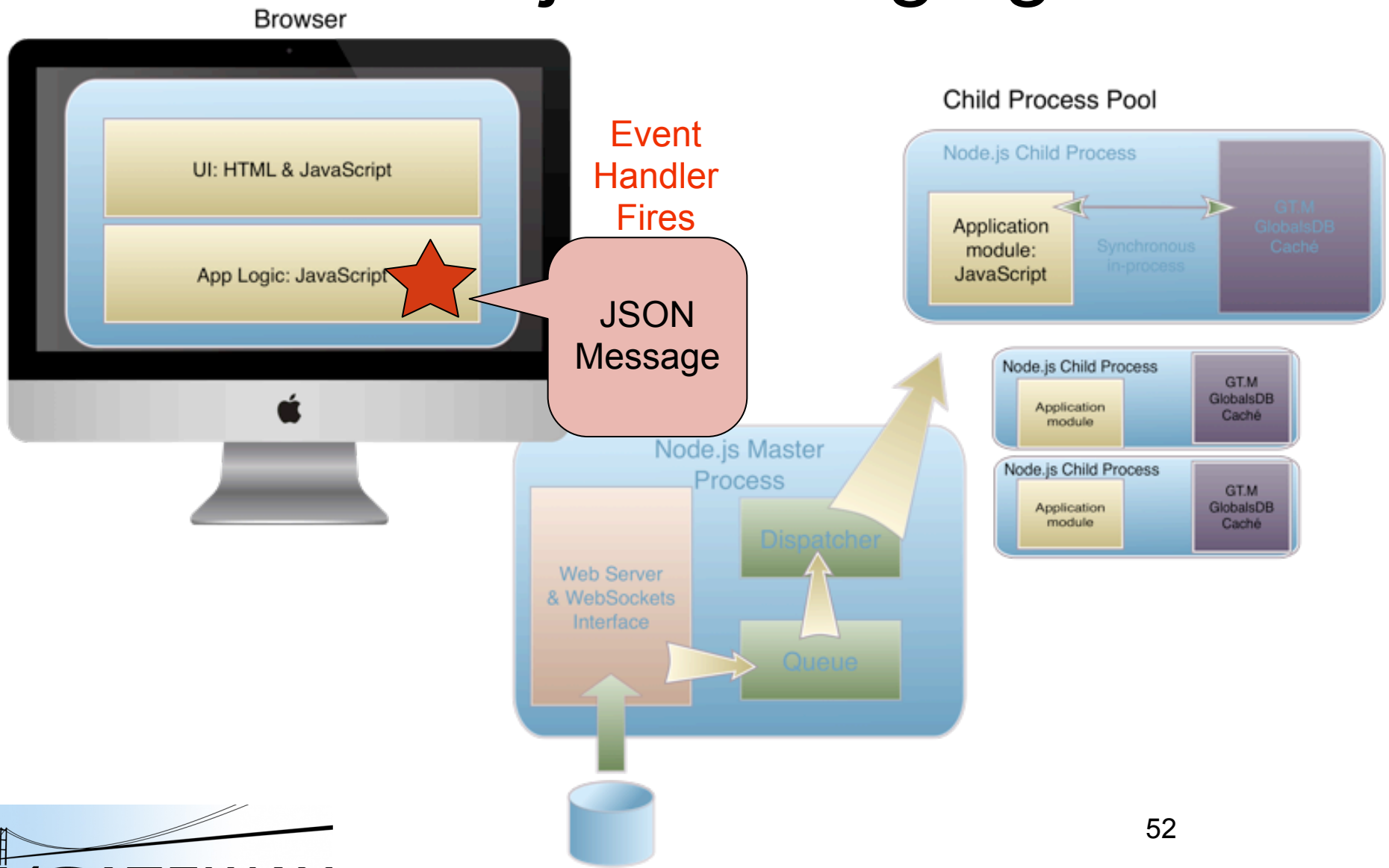


M/GATEWAY DEVELOPMENTS

EWD.js Messaging

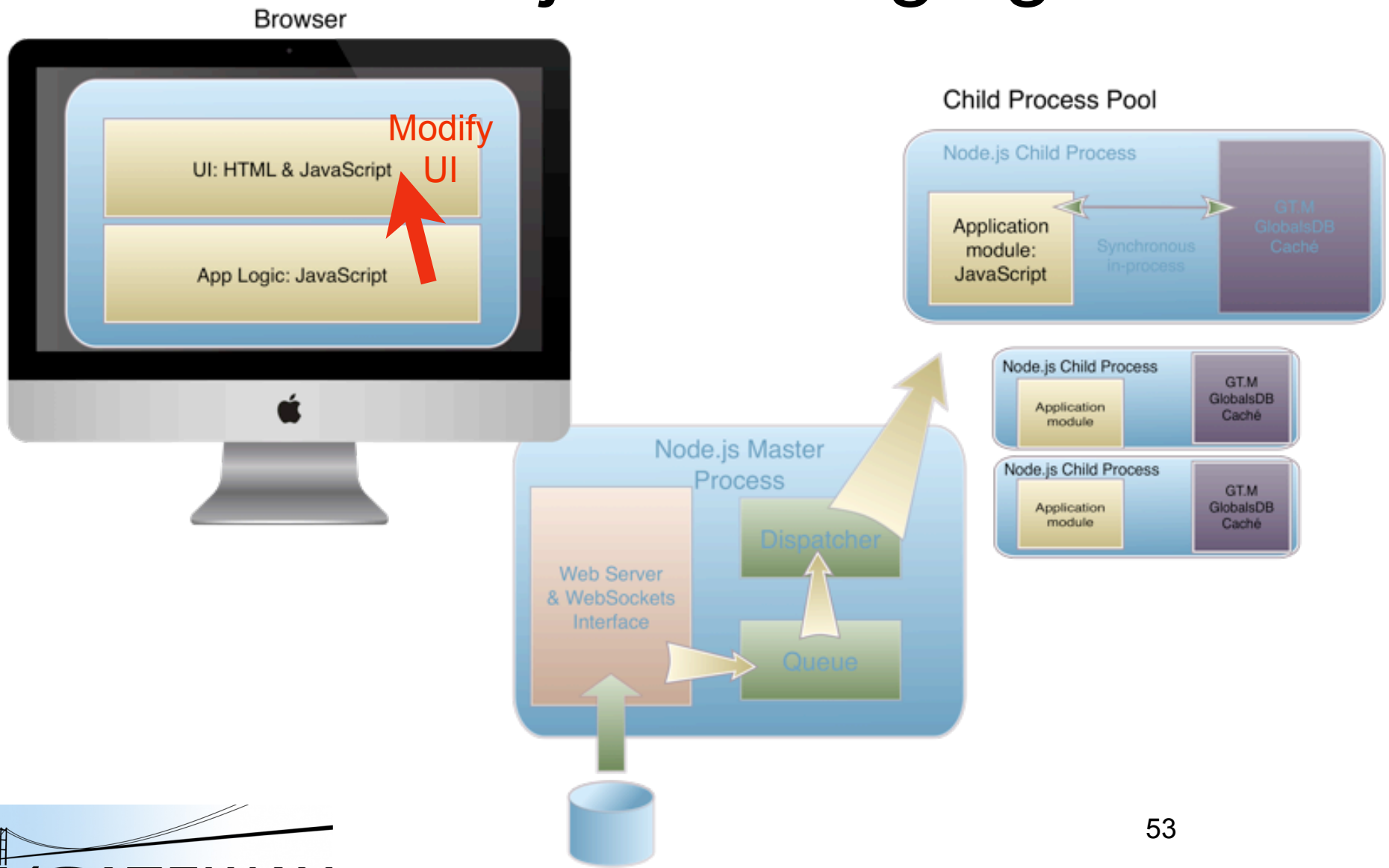


EWD.js Messaging



M/GATEWAY DEVELOPMENTS

EWD.js Messaging



M/GATEWAY DEVELOPMENTS

Send message from browser

```
EWD.sockets.sendMessage({
  type: "sendHelloWorld",
  params: {
    text: 'Hello World!',
    sender: 'Rob',
    date: new Date().toUTCString()
  }
});
```

Send message from browser

```
EWD.sockets.sendMessage({
  type: "sendHelloWorld",
  params: {
    text: 'Hello World!',
    sender: 'Rob',
    date: new Date().toUTCString()
  }
});
```

User-defined type

Send message from browser

```
EWD.sockets.sendMessage({
  type: "sendHelloWorld",
  params: {
    text: 'Hello World!',
    sender: 'Rob',
    date: new Date().toUTCString()
  }
});
```

JSON payload

Back-end Module

```
module.exports = {
  onSocketMessage: function(ewd) {
    var wsMsg = ewd.webSocketMessage;
    var type = wsMsg.type;
    var params = wsMsg.params;
    var sessid = ewd.session.$('ewd_sessid')._value;
  }
};
```

Back-end Module

```
module.exports = {
  onSocketMessage: function(ewd) {
    var wsMsg = ewd.webSocketMessage;
    var type = wsMsg.type;
    var params = wsMsg.params;
    var sessid = ewd.session.$('ewd_sessid')._value;
  }
};
```

Message Handler: fires whenever a message is received

Back-end Module

```
module.exports = {
  onSocketMessage: function(ewd) {
    var wsMsg = ewd.webSocketMessage;
    var type = wsMsg.type;
    var payload = wsMsg.params;
    var sessid = ewd.session.$('ewd_sessid')._value;
    if (type === 'sendHelloWorld') {
      // do whatever is required with payload
      return {received: true};
    }
  }
};
```

Handle the message we sent from browser

Back-end Module

```
module.exports = {
  onSocketMessage: function(ewd) {
    var wsMsg = ewd.webSocketMessage;
    var type = wsMsg.type;
    if (type === 'sendHelloWorld') {
      var savedMsg = new ewd.mumps.GlobalNode('myMessage', []);
      savedMsg._setDocument(wsMsg);
      return {savedInto: '^myMessage'};
    }
  }
};
```

Saves the entire message into ^myMessage

Back-end Module

```
module.exports = {
  onSocketMessage: function(ewd) {
    var wsMsg = ewd.webSocketMessage;
    var type = wsMsg.type;
    if (type === 'sendHelloWorld') {
      var savedMsg = new ewd.mumps.GlobalNode('myMessage', []);
      savedMsg._setDocument(wsMsg);
      return {savedInto: '^myMessage'};
    }
  }
};
```

Returns a *sendHelloWorld* message back to browser

Browser-side Handler

```
EWD.onSocketMessage = function(messageObj) {
  if (messageObj.type === 'sendHelloWorld') {
    var text = 'Your message was successfully saved into ' +
       messageObj.message.savedInto;
    document.getElementById('response').innerHTML = text;
    setTimeout(function() {
      document.getElementById('response').innerHTML = '';
    },2000);
  }
};
```

Built-in Event Handler function

Browser-side Handler

```
EWD.onSocketMessage = function(messageObj) {
  if (messageObj.type === 'sendHelloWorld') {
    var text = 'Your message was successfully saved into ' +
       messageObj.message.savedInto;
    document.getElementById('response').innerHTML = text;
    setTimeout(function() {
      document.getElementById('response').innerHTML = '';
    },2000);
  }
};
```

Just like at the back-end!

Browser-side Handler

```
EWD.onSocketMessage = function(messageObj) {
  if (messageObj.type === 'sendHelloWorld') {
    var text = 'Your message was successfully saved into ' +
       messageObj.message.savedInto;
    document.getElementById('response').innerHTML = text;
    setTimeout(function() {
      document.getElementById('response').innerHTML = '';
    },2000);
  }
};
```

Modify the UI

No Polling!

- With WebSockets, the back-end can send a message *at any time* to:
  - a specific browser
  - all browsers running a specific EWD.js application
  - all currently-connected browsers

Back-end sending a message

```
var savedMsg = new ewd.mumps.GlobalNode('myMessage', []);

ewd.sendWebSocketMsg({
  type: 'savedMessage',
  message: savedMsg._getDocument()
});
```

Invoking Mumps code

- Can invoke functions from within the back-end JavaScript module:

```
var result = ewd.mumps.function('getPatientVitals^MyEHR',
                                params.patientId,
                                params.date);
```

Invoking Mumps code

- Can invoke functions from within the back-end JavaScript module:

```
var result = ewd.mumps.function('getPatientVitals^MyEHR',
                                params.patientId,
                                params.date);
```

This is the equivalent of the Mumps code:

```
set result=$$getPatientVitals^MyEHR(patientId,date)
```

Invoking Mumps code

- Can invoke functions from within the back-end JavaScript module:

```
var result = ewd.mumps.function('getPatientVitals^MyEHR',
                                params.patientId,
                                params.date);
```

This is the equivalent of the Mumps code:

```
set result=$$getPatientVitals^MyEHR(patientId,date)
```

Then use *_getDocument()* to retrieve Vitals from Global to corresponding JSON

Built-in secured Web Services

- Any back-end JavaScript method can be exposed as a JSON Web Service
- Access is automatically secured
  - HMAC-SHA256 digital signatures required for every HTTP request
  - The same security used by Amazon Web Services
- Lightweight peer-to-peer access between EWD.js systems

Example Web Service

```
webServiceExample: function(ewd) {
  var patient = new ewd.mumps.GlobalNode('CLPPats', [ewd.query.id]);
  if (!patient._exists) return {error: 'Patient ' + ewd.query.id + ' does not exist'};
  return patient._getDocument();
}
```

```
https://192.168.1.89:8080/json/demo/webServiceExample?
  id=1233&
  accessId=rob12kjh1i23&
  timestamp=Wed, 19 Jun 2013 14:14:35 GMT&
  signature=P0blakNehj2TkuadxbKRsIgJCGIhY1EvntJdSce5XvQ=
```

Node.js EWD.js Web Service client:

npm install ewdliteclient

Example Web Service

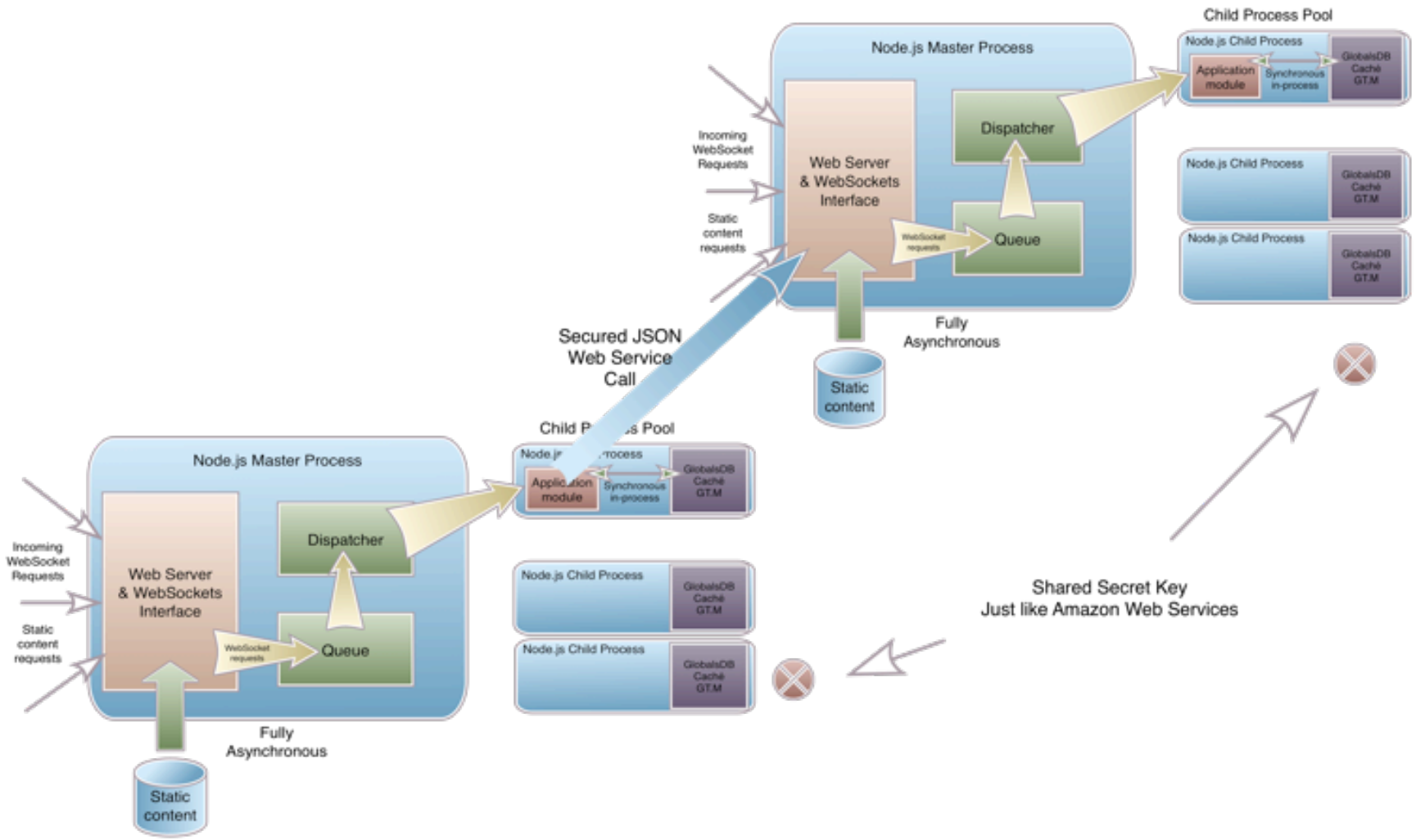
```
webServiceExample: function(ewd) {
  var patient = new ewd.mumps.GlobalNode('CLPPats', [ewd.query.id]);
  if (!patient._exists) return {error: 'Patient ' + ewd.query.id + ' does not exist'};
  return patient._getDocument();
}
```

```
https://192.168.1.89:8080/json/demo/webServiceExample?
  id=1233&
  accessId=rob12kjh1i23&
  timestamp=Wed, 19 Jun 2013 14:14:35 GMT&
  signature=P0blakNehj2TkuadxbKRsIgJCGIhY1EvntJdSce5XvQ=
```

Node.js EWD.js Web Service client:
npm install ewdliteclient

The perfect architecture to support VSA

Secured Linked Systems



M/GATEWAY DEVELOPMENTS

Node.js Custom Events

addMedication({params})

Node.js Custom Events

```
addMedication({params})
```

```
addMedication = function(params) {
... code for adding medication

ewd.emit('audit', {auditParams};
ewd.emit('stockControl', {stockParams});
};
```

Node.js Custom Events

```
addMedication({params})
```

```
addMedication = function(params) {
... code for adding medication

ewd.emit('audit', {auditParams};
ewd.emit('stockControl', {stockParams});
};
```

```
ewd.on('audit', function(params) {
  .... code for adding audit record

  possibly via WebService to remote
  system
};
```

Node.js Custom Events

```
addMedication({params})
```

```
addMedication = function(params) {
... code for adding medication

ewd.emit('audit', {auditParams};
ewd.emit('stockControl', {stockParams});
};
```

```
ewd.on('audit', function(params) {
  .... code for adding audit record
};
```

```
ewd.on('stockControl', function(params {
  .... code for changing stock record
};
```

EWD.js v “classic” EWD

- 100% JavaScript
  - no direct use of Mumps code
  - no Mumps routines to install and configure
  - works with InterSystems’ GlobalsDB
- Static pages of HTML and JavaScript
  - not an “server pages” technology
  - no compilation stage
- Not a tag-based development environment
- Works with any and all JavaScript frameworks, or hand-crafted HTML
- No HTTP / Ajax

EWD.js requirements

- Ideally browsers that support HTML5 WebSockets
  - however, EWD.js uses Node.js socket.io library
    - emulates websockets using other techniques if not available
    - even works with old versions of Internet Explorer!

Licensing & Availability

- Apache 2
- https://github.com/robtweed/ewdGateway2
- Installing on Node.js:
  - *npm install ewdgateway2*

Getting Started

- http://gradvs1.mgateway.com/download/EWDjs.pdf
- dEWDrop VM (http://www.fourthwatchsoftware.com)
- Mike Clayton's Ubuntu Installer
  - Node.js
  - EWD.js
  - GlobalsDB
- Raspberry Pi!
- Training Course:
  - WorldVistA meeting, Sacramento: Jan 2014
  - UK?

EWD.js

Rob Tweed
M/Gateway Developments Ltd

Twitter: @rtweed
Email: rtweed@mgateway.com