Web Based Computer Music UIs with Mongrel2 and Harbinger

Abram Hindle

Toronto Perl Mongers

abram.hindle@softwareprocess.es

Introduction

- What if we could use familiar interfaces to make music
- Could we co-opt other software and events to produce musical events?
- Could we even use web interfaces and use java-script UIs?

Imagine

- Extending your spread sheet into a soundboard or drum kit
- Turning your paint program into a violin
- Sonifying a boring website like eBay
- Taking a flash game, a SVG program, or HTML5 program and using it as computer music input.

Lets Hijack Something

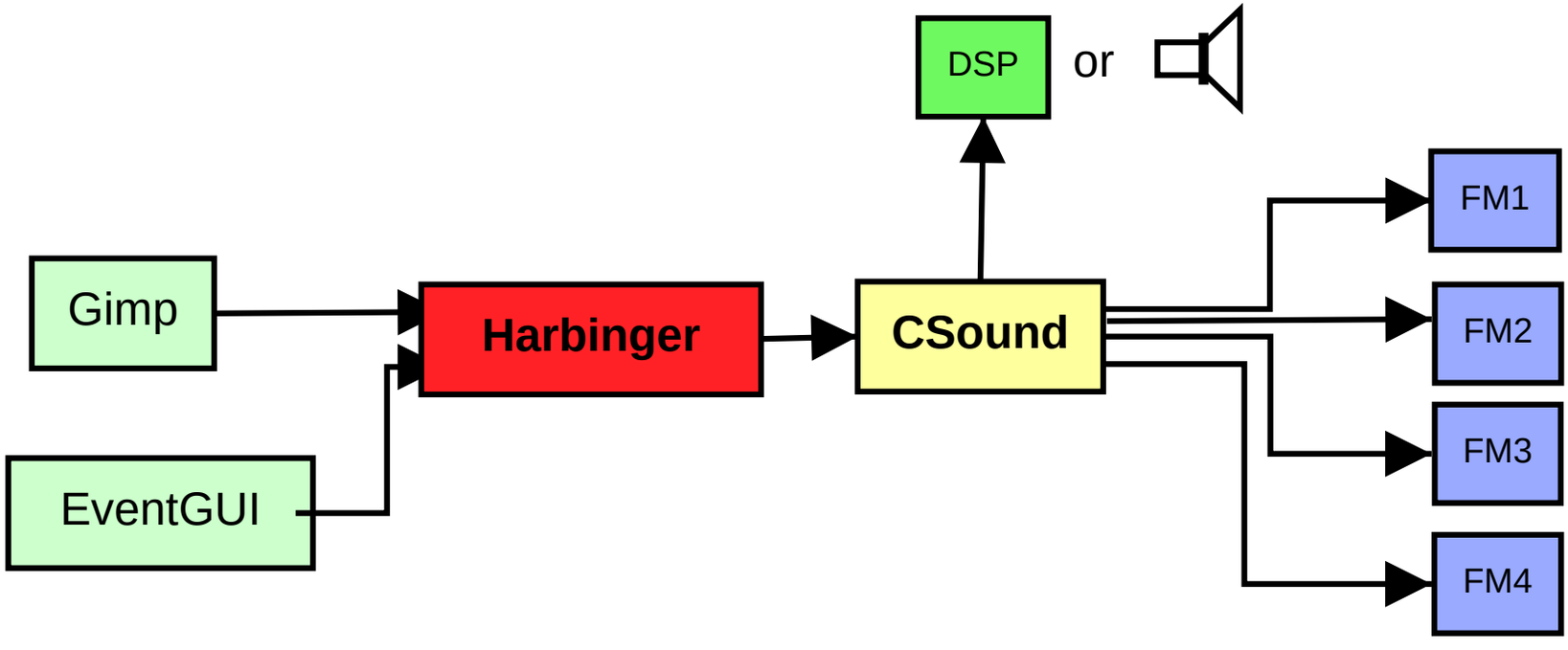
- JS1k (js1k.org) demos (now)
- Javascript programs (now)
- Websites

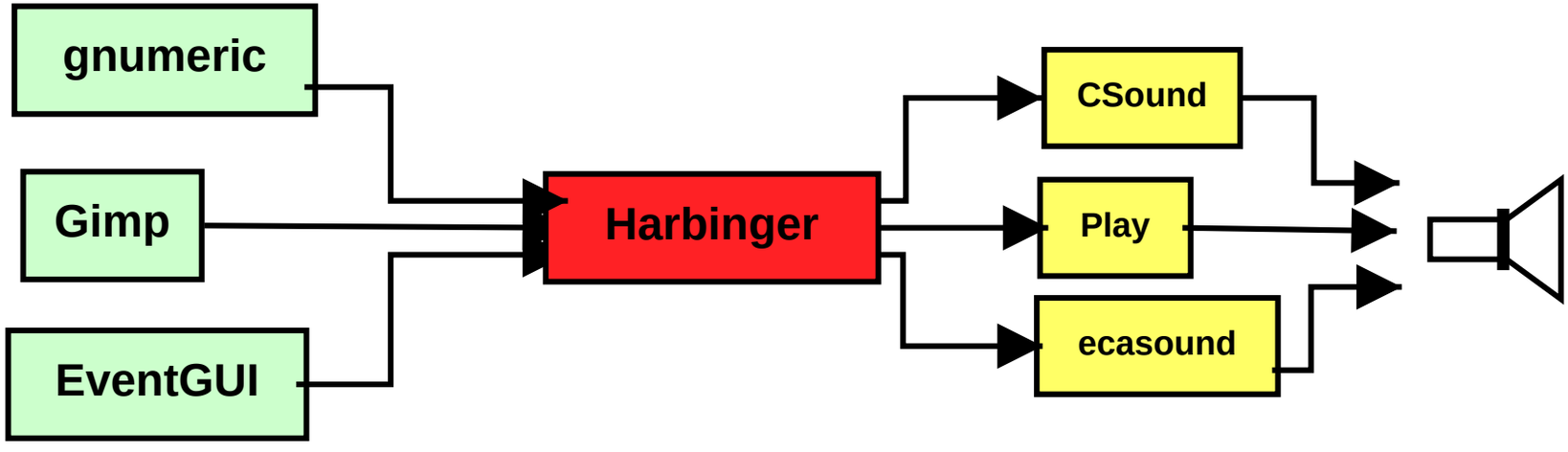
What am I talking about

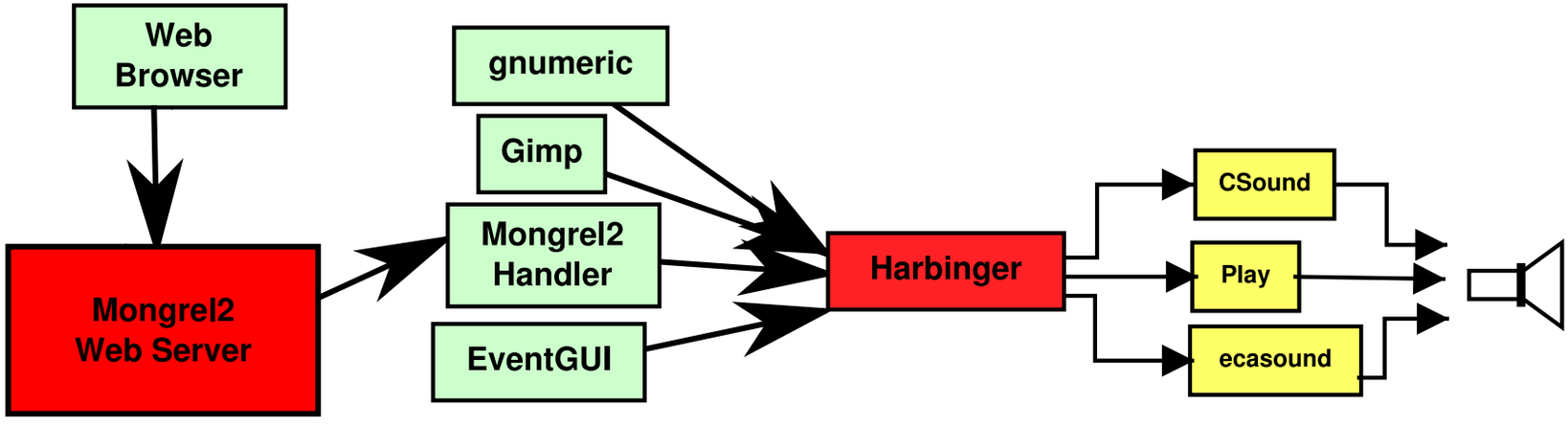
- Abram will now show a demo of filler creep sonified

What was that?

- Java-script game using Canvas
- Instrumented to send XMLHttpRequests to a mongrel2 server
- Mongrel handler communicates to Harbinger
  - Who munges messages, massages them and hands them to:
- CSound receives messages from Harbinger.


DSP
or
FM1
Gimp
Harbinger
CSound
FM2
FM3
EventGUI
FM4


gnumeric
Gimp
EventGUI
Harbinger
CSound
Play
ecasound


Web Browser
Mongrel2 Web Server
gnumeric
Gimp
Mongrel2 Handler
EventGUI
Harbinger
CSound
Play
ecasound

Ok.. but how do we communicate from the Web?

- XMLHttpRequest
  - AJAX
  - Send a background request from a web UI
- Mongrel2 and Mongrel2 Handler receives it
  - Pass off to Harbinger

Harbinger

- send connectionless messages encoded in plain text
  - UDP
- middle man routes these messages, and massages them
  - Add code to the middle man to filter messages
- middle man passes messages on to other programs

Why not OSC?

- OSC wasn’t around or popular the time
- This is very simple
- This is very easy
- I can do more than OSC
- No errors if there is no one receiving the message
- Reduce dependencies.

Guts and Bolts

- Let’s get dirty, we’re going to discuss and install:
  - ZeroMQ
  - Mongrel2
  - HTTP Handler
  - Java-script instrumentation

ZeroMQ

- Fast messaging
- Wide variety of options
- `http://zeromq.com/`

```
wget \
 http://www.zeromq.org/local--files/area
tar zxvf zeromq*gz
cd zermq-*
./configure && make && sudo make instal
```

ZeroMQ

- Claims great cross language/platform network performance
- Asynchronous and easily event driven
  - large message sizes supported
- Has Perl support `cpan ZeroMQ`
  - you might have to skip the tests
- We will come back to ZeroMQ
- Appropriate for local, cross-core, cross-interconnect and cross-network communication

Mongrel2

- `http://mongrel2.org`
- Based on popular ruby mongrel web-server
  - Rails app server
- Aims to be a stable and fast web-server
- Not really general purpose IMHO
  - meant for application specialization
- Seems fast to me 2.8ms response (1000 requests in 2.8 seconds in Perl) (beta0)
- Not totally stable yet

Mongrel2

- Zed Shaw's baby and he seems to care about it.
- Too Pythony right now
  - Uses SQLite for configuration BUT the config examples are all python and DSLish.
- Requires ZeroMQ and a lot of python
  - Getting python packages sucks.

Mongrel2 Config

- Control via m2sh, but first we need to make a config

```
main = Server(
  uuid="cd27a9c2-386b-4b51-b21e-3e5635a94551",
  access_log="/logs/access.log",
  error_log="/logs/error.log",
  chroot="./",
  default_host="localhost",
  pid_file="/run/mongrel2.pid",
  port=6767,
  hosts = [ Host(name="localhost", routes=
       myroute) ]
)
commit([main])
```

Mongrel2 Routes

- Routes are the paths that you react to

```
myroute = {
  r’/demos/’: Dir(base=’harbinger-demos/’,
      index_file=’index.html’,
                  default_ctype=’text/plain’),
  r’/harbinger’: harbinger,
}
```

Mongrel2 Handlers

- mongrel will listen on ports for the handler
  - Need a send port
  - Need a recv port
- Handler needs an UUID ident (128bit)
- Our handler - the app

```
harbinger = Handler(
  send_spec='tcp://127.0.0.1:9967',
  send_ident='f8144414-ad7a-11df-9185-001
      bfce70aad',
  recv_spec='tcp://127.0.0.1:9966',recv_ident='
      ')
```

Mongrel2

- Ok fine but how do we communicate with mongrel
  - how do we serve an app?
    - handlers!
- ZeroMQ comes back, to communicate to mongrel you use JSOn (yuck) and ZeroMQ (yeah!)
- Show http.py
- Show http.pl

ZeroMQ and Perl

- ZeroMQ module
  - Need a context object
    - DON’T LOSE IT
    - Don’t garbage collect it til you’re done!

ZeroMQ Perl Example

```
my $sender_id = 'f8144414-ad7a-11df-9185'.
  '-001bfce70aad';
my $sub_addr = "tcp://127.0.0.1:9997";
my $pub_addr = "tcp://127.0.0.1:9996";
my $cxt = ZeroMQ::Context->new;
my $req_socket = ZeroMQ::Socket->new($cxt,
  ZMQ_UPSTREAM);
$req_socket->connect($sub_addr);
my $resp_socket = ZeroMQ::Socket->new($cxt,
  ZMQ_PUB);
$resp_socket->connect($pub_addr);
$resp_socket->setsockopt( ZMQ_IDENTITY,
  $sender_id );
while(my $req = $req_socket->recv) ...
```

Mongrel2 http.pl

- Skip http.py because you don’t need it
- Go through http.pl
- Show Mongrel2.pm

Harbinger and Mongrel

- Now using Mongrel2.pm we make a handler
  - To transmit messages to the instrument
  - Show http_harbinger.pl

Java-script XMLHttpRequest stuff

- You have to send messages back

```
function harb(msg) {
  var xhr = new XMLHttpRequest();
  xhr.open("POST",
    "http://localhost:6767/harbinger",
    true);
  xhr.send( JSON.stringify(
    { "program":"filler",
      "id":666, "dest":"",
      "msg":msg }) );
}
```

Java-script XMLHttpRequest stuff

- You might have to sample results instead of sending updates
  - `setinterval(hardSender,100);`
- You might have to queue things up or split them up
- You might have to change mongrel settings to allow for larger posts or gets

Harbinger instrument

- Show example instrument
- demo example instrument

How can you use Mongrel?

- Grab my Mongrel2.pm `http://ur1.ca/17bmn`
- Grab my http_harbinger.pl
  `http://ur1.ca/1a9zm`
- Gut the http_harbinger.pl and put your code in there
- Use lestratt's Plack mongrel2 handler
  `http://ur1.ca/1aa17`

Caveats

- Mongrel2 seems to be evolving quickly
- Bugs pop up and go away, so watch out
- There are lots of settings to fiddle with in the manual
  - if you get strange behaviour maybe your buffers are not big enough.

Future Work

- Fish or Cut Bait?
  - Work on music
  - Work on tools?
  - Stream MP3s back?

Conclusions

- Take home:
  - leverage the UIs of other software to produce music
  - Perl is the great glue that holds systems together
  - Mongrel2 is fast enough for this job
  - ZeroMQ is pretty interesting
- Get this presentation and code at:

```
http://github.com/abramhindle/mongrel2-musical-relay

http://softwareprocess.es/index.cgi/WebBasedComputerMusic
```

Garbage notes

- Most of this is in mongrel2*/doc/manual/installing.tex

```
sudo aptitude install uuid-dev
sudo aptitude install csound

wget http://mongrel2.org/static/downloads/mongrel2-1.0beta3.tar.bz2
  (get the latest)
http://zeromq.org
wget http://pypi.python.org/packages/source/d/distribute/distribute
```

```
        -0.6.14.tar.gz#md5=
        ac607e05682116c06383b27a15e2db90
    wget http://pypi.python.org/packages/
        source/p/pip/pip-0.7.2.tar.gz#md5=
        cfe73090aaa0d3b0c104179a627859d1
    tar zxvf distribute-0.6.14.tar.gz
    cd distribute-0.6.14
    sudo python setup.py install
    cd ..
    cd pip-0.7.2
    sudo python setup.py install
    cd
```

```
wget http://www.zeromq.org/local--files
    /area:download/zeromq-2.0.8.tar.gz
tar zxvf zeromg-*.tar.gz
cd zeromq-2.0.8
./configure && make && sudo make
    install
cd ..
sudo cpan ZeroMQ # this has a lot of
    deps seems like a real pro module
# otherwise
# git clone http://github.com/tsee/
    ZeroMQ-Perl.git
```

```
# wget http://search.cpan.org/CPAN/
    authors/id/D/DM/DMAKI/ZeroMQ-0.02.
    tar.gz
# cd ZeroMQ-0.02
# perl Makefile.pl
# make
# make test # these failed for me!
# sudo make install
git clone http://github.com/zeromq/
    pyzmq.git
cd pyzmq
sudo python setup.py install
```

```
cd ..
# for examples
sudo easy_install web.py
cd mongrel2_2010*
make all
sudo make install
#test it
m2sh
cd ..
git clone http://github.com/abramhindle
    /mongrel2-musical-relay.git
cd mongrel2-musical-relay
```