The Team

Steve
EE

Eric
EE

Jennifer
SQL

Cris
Python
Roadmap

• Introduction
  • Solution Overview
  • Energy Efficiency 101
  • Site Tour

• Under the Hood
  • Basic Machine Configuration (AMIs)
  • Software Configuration
  • Database Structure
  • The Web Application
The Solution

AWS EC2

- m2.xlarge (high-memory)
  - Zope Object Database
  - PostgreSQL Database
  - ZEO Server
  - PG Bouncer
  - monitoring

- c1.medium (high-CPU)
  - Zope Client
  - Apache
  - Django (mod_wsgi)
  - monitoring

Database Applications

Front-End Applications

VCS (beanstalk)

Issue Tracking (FogBugz)

zc.buildout

Development Machine
mr.developer

m1.small

Monitoring Dashboard

User's Browser
jQuery
OpenLayers

User

Developer
Why CA Energy Efficiency?
World Leader

- 1.5 Million Jobs
- $45 Billion in Payroll
How does that compare to the rest of the U.S?
California...

- ... spends billions less on Electricity
- ... has a ratio of GDP to Energy Consumed 68% higher
- ... residents, per capita, pay lower utility bills
"Imagine where the country could be if it were as efficient as California."
- F. Noel Perry, venture capitalist, founder of Next 10
Energy Efficiency and EMV
How Does CA do Energy Efficiency?

• CPUC established EE program cycle length and funding level

• CPUC provides $ to utilities to implement EE Programs

• Utilities report program accomplishments to CPUC
How Does CA do Energy Efficiency?

- CPUC hires independent EMV contractors to independently evaluate utility savings claims.
- CPUC awards $ to the utilities based on savings accomplishments.
- CPUC releases data publicly via website.
Rinse and Repeat
What is OpenEMV?
The open source for measured performance
Energy efficiency decisions require data on the performance of energy efficiency measures. Measured energy savings data from program evaluation, measurement and verification (EM&V) studies exist in many contexts. OpenEMV's purpose is to make available all public and open EM&V data to support energy efficiency policy and business decision making.

Tell me more about OpenEMV

What data do we have?
More than 3.7 million records of energy efficiency data
The California public dataset contains over 3.7 million records, representing efficiency projects and measure installations across the state. OpenEMV presents these data in an interactive format. You can use our tools to gain a better understanding of the magnitude of energy and cost savings, the geographic distribution of the savings, and the kinds of technologies and buildings involved.

Show me the data

Why Open?
Public data and engaged community
Our goal is to serve both as a community space and as a source of technical data. By adopting the open source model, we assure that any party can contribute to and benefit from the data.

Tell me more about open data

Why EM&V?
Solid data for solid decisions
Across the globe, enormous amounts of money are being spent collecting energy efficiency data. If all these data were openly available, they could be useful to efficiency program designers, policy makers, and other interested parties. A single, shared repository and analysis tools will improve the availability of measure performance data, lower program costs and provide support for better decisions.

Tell me more about EM&V

Why California?
The largest efficiency program evaluation effort in history
In 2006-2008, California state government and utility companies worked together to generate an unprecedented dataset on energy efficiency and conservation measure performance. The data were made public as a matter of policy. OpenEMV is using this dataset as an example of the types of analyses that can performed by anyone, anywhere, using open source tools.

Tell me more about California

Upcoming
 SCALE 9x
California Energy Efficiency Program Data

Show me a map of **firstyear evaluated kW savings by State House District**

Legend

- 0 – 15,100 kW
- 15,100 – 30,100 kW
- 30,100 – 45,100 kW
- 45,100 – 60,100 kW
- 60,100 – 75,100 kW
- 75,100+ kW

Download data: CSV, KML
California Energy Efficiency Program Data

Show me a map of **lifecycle evaluated kWh savings** by **State Senate District**

compared to **lifecycle total investment cost** by **State Senate District**

Legend
- 0 – 613,600,000 kWh
- 613,600,000 – 1,200,000,000 kWh
- 1,200,000,000 – 1,800,000,000 kWh
- 1,800,000,000 – 2,400,000,000 kWh
- 2,400,000,000 – 3,000,000,000 kWh
- 3,000,000,000+ kWh

Download data: CSV, KML

Legend
- $0 – $32,600,000
- $32,600,000 – $62,500,000
- $62,500,000 – $92,400,000
- $92,400,000 – $122,300,000
- $122,300,000 – $152,100,000
- $152,100,000+

Download data: CSV, KML
Roadmap

- Introduction
- Solution Overview
- Energy Efficiency 101
- Site Tour

- Under the Hood
  - Basic Machine Configuration (AMIs)
  - Software Configuration
  - Database Structure
  - The Web Application
Roadmap

- Introduction
- Solution Overview
- Energy Efficiency 101
- Site Tour
- Under the Hood
  - Basic Machine Configuration (AMls)
  - Software Configuration
  - Database Structure
  - The Web Application
AMI
(Amazon Machine Image)
Custom DB AMI

- **m2.xlarge** (high-memory)
- **Database Applications**
  - Zope Object Database
  - PostgreSQL Database
  - ZEO Server
  - PG Bouncer

- **monitoring**
Custom DB AMI
Custom DB AMI

- Ubuntu 10.10 (Maverick)
Custom DB AMI

- Ubuntu 10.10 (Maverick)
- 64-bit
Custom DB AMI

- Ubuntu 10.10 (Maverick)
- 64-bit
- Kernel Settings
Custom DB AMI

- m2.xlarge (high-memory)

Database Applications

- PostgreSQL Database
- PG Bouncer
- monitoring

- Ubuntu 10.10 (Maverick)
- 64-bit
- Kernel Settings

```
kernel.shmmax = 8GB
kernel.shmall = 8GB/4096
/etc/sysctl.conf
```
Custom DB AMI

- Ubuntu 10.10 (Maverick)
- 64-bit
- Kernel Settings
- Spatial Database
Custom DB AMI

- Ubuntu 10.10 (Maverick)
- 64-bit
- Kernel Settings
- Spatial Database

apt-get install...
PostgreSQL 9.0.3
Geos 3.2.2
GDAL 1.7.0
manual install
postgis 1.5.2
Custom DB AMI

- Ubuntu 10.10 (Maverick)
- 64-bit
- Kernel Settings
- Spatial Database
- Connection Pooling

Diagram:
- m2.xlarge (high-memory)
- Database Applications
- PostgreSQL Database
- PG Bouncer
- monitoring
Custom DB AMI

- Ubuntu 10.10 (Maverick)
- 64-bit
- Kernel Settings
- Spatial Database
- Connection Pooling

apt-get install... pgbouncer 1.3.3
Custom DB AMI

- m2.xlarge (high-memory)
- PostgreSQL Database
- PG Bouncer
- monitoring
- Database Applications
- Ubuntu 10.10 (Maverick)
- 64-bit
- Kernel Settings
- Spatial Database
- Connection Pooling
- Monitoring
Custom DB AMI

- Ubuntu 10.10 (Maverick)
- 64-bit
- Kernel Settings
- Spatial Database
- Connection Pooling
- Monitoring

apt-get install...
monit
munin-node
Custom Web App AMI

- c1.medium (high-CPU)
- Front-End Applications
  - Zope Client
  - Zope Client
  - Zope Client
- Apache
  - Django (mod_wsgi)
- monitoring
Custom Web App AMI

c1.medium (high-CPU)

Front-End Applications

Apache

monitoring
Custom Web App AMI

- Ubuntu 10.10 (Maverick)
Custom Web App AMI

- Ubuntu 10.10 (Maverick)
- 32-bit
Custom Web App AMI

- Ubuntu 10.10 (Maverick)
- 32-bit
- Apache2
Custom Web App AMI

- Ubuntu 10.10 (Maverick)
- 32-bit
- Apache2
- mod_wsgi
Custom Web App AMI

- Ubuntu 10.10 (Maverick)
- 32-bit
- Apache2
- mod_wsgi
- Monitoring
We’ll make these public
We’ll make these public

– once I’ve polished them up a bit :)

OPEN EMV
Additional Software
Additional Software

- Back-End
Additional Software

- Back-End
- ZODB
Additional Software

- Back-End
- ZODB
- ZEO Server
Additional Software

- Back-End
  - ZODB
  - ZEO Server
- Front-End
Additional Software

- **Back-End**
  - ZODB
  - ZEO Server

- **Front-End**
  - Django
Additional Software

- Back-End
  - ZODB
  - ZEO Server

- Front-End
  - Django
  - Plone
Additional Software

- **Back-End**
  - ZODB
  - ZEO Server

- **Front-End**
  - Django
  - Plone
  - Add-Ons
Additional Software

- Back-End
  - ZODB
  - ZEO Server

- Front-End
  - Django
  - Plone
    - Add-Ons
  - Zope Clients
And Configuration

• Back-End
  • ZODB
  • ZEO Server

• Front-End
  • Django
  • Plone
  • Add-Ons
  • Zope Clients
And Configuration

- Back-End
  - ZODB
  - ZEO Server

- Front-End
  - Django
  - Plone
    - Add-Ons
    - Zope Clients
And Configuration

- Back-End
  - ZODB
  - ZEO Server
- Front-End
  - Django
  - Plone
  - Add-Ons
  - Zope Clients
- PostgreSQL
And Configuration

- Back-End
  - ZODB
  - ZEO Server
- Front-End
  - Django
  - Plone
  - Add-Ons
  - Zope Clients
- PostgreSQL
And Configuration

- Back-End
  - ZODB
  - ZEO Server
- Front-End
  - Django
  - Plone
  - Add-Ons
  - Zope Clients
- PostgreSQL
- Apache
And Configuration

• Back-End
  • ZODB
  • ZEO Server

• Front-End
  • Django
  • Plone
  • Add-Ons
  • Zope Clients

• PostgreSQL
  • Apache
And Configuration

- Back-End
  - ZODB
  - ZEO Server
- Front-End
  - Django
  - Plone
  - Add-Ons
  - Zope Clients

Databases:
- PostgreSQL

Monitors:
- Monit

Web Servers:
- Apache
And Configuration

- **Back-End**
  - ZODB
  - ZEO Server
- **Front-End**
  - Django
  - Plone
- **Add-Ons**
- **Zope Clients**

Replicate On Development Machine(s)
This is a job for Superman!
This is a job for Buildout!
Wait – what?
zc.buildout

www.buildout.org
developed for ZoPE
used extensively by Plone®
also gaining traction with

django

GROK

Pyramid
not limited to python
dependencies
dependencies

$ apt-get install python-setuptools
dependencies

$ apt-get install python-setuptools
$ easy-install distribute
dependencies

$ apt-get install python-setuptools
$ easy-install distribute
$ apt-get install python-dev
The Basics
The Basics

$ ls basic-buildout
bootstrap.py
buildout.cfg
The Basics
The Basics

$ more buildout.cfg

[buildout]
part = foo

[foo]
recipe = collective.recipe.cmd
cmd =
    echo foo
Build It

```bash
fiver:basic-buildout cewing$ python bootstrap.py
install_dir /var/folders/BK/BK9HvlYeFsaqGWLX1x8Mxk+++TI/-Tmp-/tmpm1quqi
Creating directory '/Users/cewing/basic-buildout/bin'.
Creating directory '/Users/cewing/basic-buildout/parts'.
Creating directory '/Users/cewing/basic-buildout/develop-eggs'.
Generated script '/Users/cewing/basic-buildout/bin/buildout'.
fiver:basic-buildout cewing$
```
Run It

```
bin/buildout

/Users/cing/basic-buildout/parts/buildout/site.py:262: UserWarning: Module pkg_resources was already imported from /System/Library/Frameworks/Python.framework/Versions/2.6/Extras/lib/python/pkg_resources.pyc, but /Users/cing/.buildout/eggs/distribute-0.6.14-py2.6.egg is being added to sys.path
  import pkg_resources
/Users/cing/basic-buildout/parts/buildout/site.py:262: UserWarning: Module site was already imported from /Users/cing/basic-buildout/parts/buildout/site.pyc, but /Users/cing/.buildout/eggs/distribute-0.6.14-py2.6.egg is being added to sys.path
  import pkg_resources
Installing foo.
foo
```
let’s scale this up
lots of cfg, and more
[buildout]
extensions =
    mr.developer
    buildout.threatlevel
    buildout.dumppickedversions

sources = sources
auto-checkout =
    sdm.openemv_policy
    openemv_maps
    openemv_theme

# Change the number here to change the version of Plone being used
extends =
    http://dist.plone.org/release/4.0.3/versions.cfg

versions = versions

# Add additional egg download sources here. dist.plone.org contains archives
# of Plone packages.
find-links =
    http://dist.plone.org/release/4.0.3
    http://dist.plone.org/thirdparty
    http://dist.plone.org
    http://download.zope.org/ppix/
    http://download.zope.org/distribution/
    http://effbot.org/downloads
    http://dist.repoze.org/

# Add additional eggs here
plone-eggs =
    Plone
    PIL==1.1.6
    Products.Scrawl
    Products.ContentWellPortlets
    Products.Collage
    collective.collage.portlets
    http://mỳ.opendev.org/openemv_policy

# we will not need to use the hotfix once we've upgraded past 4.0.3,
# but we need it now
    Products.PloneHotfix20110720

plone-dev-eggs =
    plone.reload
    teamrubber.theoracle == 0.0.5
### Parts for building spatial PostgresQL database (dev only)

```
[postgresql]
recipe = zc.recipe.cmmi
extra-options =
  --with-readline
  --enable-thread-safety

[proj]
recipe = hexagonit.recipe.cmmi
url = http://download.osgeo.org/proj/proj-4.7.0.tar.gz

[geos]
recipe = hexagonit.recipe.cmmi
url = http://download.osgeo.org/geos/geos-3.2.2.tar.bz2

[gdal]
recipe = hexagonit.recipe.cmmi
url = http://download.osgeo.org/gdal/gdal-1.7.2.tar.gz
configure-options =
  --with-python
  --with-geos=${geos:location}/bin/geos-config

[postgis]
recipe = hexagonit.recipe.cmmi
url = http://postgis.refractions.net/download/postgis-1.5.2.tar.gz
configure-options =
  --with-pgsql=${postgresql:location}
  --with-pgconfig=${postgresql:location}/bin/pg_config
  --with-geos=${geos:location}
  --with-geosconfig=${geos:location}/bin/geos-config
  --with-proj=${proj:location}
  --with-projdir=${proj:location}

[init-pgsql]
recipe = collective.recipe.cmd
```
A Closer Look

```plaintext
[init-pgsql]
recipe = collective.recipe.cmd
on_install = true
on_update = true
cmds =
  mkdir -c ${buildout:directory}/var/data
  ${postgresql:location}/bin/initdb -D ${buildout:directory}/var/data -E
  UNICODE
  ${postgresql:location}/bin/pg_ctl -D ${buildout:directory}/var/data start
  sleep 5
  ${postgresql:location}/bin/createdb -E UTF8 template_postgis
  ${postgresql:location}/bin/createlang -d template_postgis plpgsql
  ${postgresql:location}/bin/psql -d template_postgis -f
  ${postgresql:location}/share/contrib/postgis-1.5/postgis.sql
  ${postgresql:location}/bin/psql -d template_postgis -f
  ${postgresql:location}/share/contrib/postgis-1.5/spatial_ref_sys.sql
  ${postgresql:location}/bin/pg_ctl -D ${buildout:directory}/var/data stop
  sleep 5
```
Tie it all together...
[buildout]
extends =
    base.cfg
    psql_parts.cfg
    django_parts.cfg
    plone_parts.cfg

parts =
    postgresql
    proj
    geos
    gdal
    postgis
    init-psql
    psql-symlinks
    psycopg2-package
    psycopg2-setup
    psycopg2-install
    django
    theme-symlinks
    django-settings
    drop-openevm
    setup-openevm
    zopepy
    zeoserver
    instance
    test
    omelette
    zopeskel

  zeoserver-address = 127.0.0.1
  zeoserver-port = 8901
  ip-address = 127.0.0.1
  instance-port = 8080
  debug-mode = on
  verbose-security = on
  deprecation-warnings = on
  django-db-user =
  django-db-passwd =
  django-db-host =
or, for deployment...
```
[buildout]
extends =
  base.cfg
  plone_parts.cfg
  admin_parts.cfg

parts =
  ploneuser-setup
  zeoserver
  zeo-init
  backup
  monit-setup
  monitconf
  monit-system
  monit-zeoserver
  monit-postgresql
  monit-pgbouncer
  postgresqlconf

zeoserver-address =
zeoserver-port = 8901
monit-port = 8100
monit-address = 127.0.0.1
effective-user = plone
host-readable-name = OpenEMV Back-End

[sources]
  sdm.openemv_policy = svn
  https://sounddatamanagement.svn.beanstalkapp.com/openemv/sdm.openemv_policy/trunk/
  openemv_maps = svn
  http://sounddatamanagement.svn.beanstalkapp.com/openemv/openemv_maps/trunk
  openemv_theme = svn
  http://sounddatamanagement.svn.beanstalkapp.com/openemv/openemv_theme/trunk

[versions]
  cns.recipe.symlink = 0.1
  collective.recipe.backup = 1.7
  collective.recipe.template = 1.8
```
Buildout Advantages

- Easy to write
Buildout Advantages

- Easy to write
- Lots of recipes, examples
Buildout Advantages

- Easy to write
- Lots of recipes, examples
- Manage complexity
Buildout Advantages

- Easy to write
- Lots of recipes, examples
- Manage complexity
  - Duplicate across platforms
Buildout Advantages

- Easy to write
- Lots of recipes, examples
- Manage complexity
  - Duplicate across platforms
  - Deploy rapidly
Buildout Advantages

- Easy to write
- Lots of recipes, examples
- Manage complexity
  - Duplicate across platforms
  - Deploy rapidly
- Versioning of software setup and config
not without issues
not without issues

- Can be network-dependent
not without issues

- Can be network-dependent
- Interconnected parts mean complexity
not without issues

- Can be network-dependent
- Interconnected parts mean complexity
- ‘extends’ can be tricky to manage
use it in joy
Data ingest

1. Download 5 MS Access databases
2. Remove special characters
3. Download CPUC website
4. Download 56 .csv files
5. Remove special characters
6. Aggregate data
7. PostgreSQL
8. Aggregate data
9. Program table
10. Measure table
11. Evaluation table

Flow diagram showing the process of data ingestion.
Data dimensions

- Program data
  - pid
  - 125:50K
  - Program groupings
  - 170:12

- Measure data
  - mid
  - pid
  - 160:3.7M
  - Measure groupings
  - 116:16

- Evaluation data
  - mid
  - 40:3.4M

- Savings calculations

- Materialized view
  - 20:200K
The Web Application

California Energy Efficiency Program Data

Show me a map of lifecycle evaluated kWh savings by State Senate District

Legend
- 0 – 613,600,000 kWh
- 613,600,000 – 1,200,000,000 kWh
- 1,200,000,000 – 1,800,000,000 kWh
- 1,800,000,000 – 2,400,000,000 kWh
- 2,400,000,000 – 3,000,000,000 kWh
- 3,000,000,000+ kWh

Download data: CSV, KML
8 Axes of Data
8 Axes of Data

- Time Frame (lifecycle or first-year)
8 Axes of Data

- Time Frame (lifecycle or first-year)
- Savings Type (but cost too)
8 Axes of Data

- Time Frame (lifecycle or first-year)
- Savings Type (but cost too)
- Year
8 Axes of Data

- Time Frame (lifecycle or first-year)
- Savings Type (but cost too)
- Year
- Utility
8 Axes of Data

- Time Frame (lifecycle or first-year)
- Savings Type (but cost too)
- Year
- Utility
- Program Group
8 Axes of Data

- Time Frame (lifecycle or first-year)
- Savings Type (but cost too)
- Year
- Utility
- Program Group
- Measure Group
8 Axes of Data

- Time Frame (lifecycle or first-year)
- Savings Type (but cost too)
- Year
- Utility
- Program Group
- Measure Group
- Building Type
8 Axes of Data

- Time Frame (lifecycle or first-year)
- Savings Type (but cost too)
- Year
- Utility
- Program Group
- Measure Group
- Building Type
- Geography
only 3 are required
8 Axes of Data

• Time Frame (lifecycle or first-year)
• Savings Type (but cost too)
• Year
• Utility
• Program Group
• Measure Group
• Building Type
• Geography
Show me the [timeframe] [savings type] by [geography]
Choices drawn from the distinct values for a specific table column

Show me the [timeframe] [savings type] by [geography]
Choices drawn from the distinct values for a specific table column

Show me the [timeframe] [savings type] by [geography]

Choices drawn from a set of table columns (not the contained values)
Show me the **timeframe** [savings type] by **geography**

Choices drawn from the distinct values for a specific table column

Choices drawn from a set of table columns (not the contained values)

Choices made up but reference table-borne data indirectly
Show me the [timeframe] [savings type] by [geography]
Show me the [timeframe] [savings type] by [geography]

for [year]
for [utility]
for [program group]
for [measure group]
for [building type]
Show me the [timeframe] [savings type] by [geography]
for [building type]
for [year]
for [utility]
for [program group]
for [measure group]
Show me the [timeframe] [savings type] by [geography] for [building type] for [year] for [utility] for [program group] for [measure group]
the sentence interface
the sentence interface

Thanks to FogBugz for the inspiration
a Django package
a Django package

A Form
a Django package

A Form
A Field
a Django package

A Form
A Field
A Widget
We’ll make this public
We’ll make this public

– once I’ve polished it up a bit :)
How ‘bout a quick Demo?
questions?
comments?