Disclaimer

This document is for informational purposes only and is subject to change at any time without notice. The information in this document is proprietary to Actian and no part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of Actian.

This document is not intended to be binding upon Actian to any particular course of business, pricing, product strategy, and/or development. Actian assumes no responsibility for errors or omissions in this document. Actian shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. Actian does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.
Ingres for the Web
Cristian Ianculovici
@crisianculovici
Agenda

Architecture
JDBC & .NET
Ingres ODBC Connectivity
- PHP
- Python
- Perl
Other Frameworks
OpenROAD JSON-RPC
Architecture
My Application

DAS Clients

ODBC Clients

Ingres Clients

Data Access Server

Communication Server

Remote Ingres Server

Ingres Database

Blah... Blah... Blah...
SURPRISING STATISTICS ABOUT BIG DATA

Wide, Wide Web
More than 570 new Websites are created every minute of the day.

Baseline
JDBC and .NET
Java and .NET

- JDBC Driver since Ingres 2.6

- .NET Data Provider
  - ASP.NET
Ingres ODBC
An application uses a connection string to connect to the database via ODBC.

- **Using a DSN**
  - `dsn=dsnname`

- **Using a DSN with a user name and password**
  - `dsn=dsnname;Uid=username;Pwd=password`

- **Connection string without a DSN**
  - `Driver={Ingres};Server=(local);Database=iidbdb`
  - `Driver={Ingres};Server=vnode_name;Database=alias;ServerType=mssql`
  - `Driver={Ingres II};Server=vnode_name;Database=alias;ServerType=ingres`

- **Dynamic Vnode Specification**
  - `DRIVER={Ingres};SERVER=@server_name,tcp_ip,II[ingres,password];PORT=II;DATABASE=userdb;`
Ingres ODBC DSN

Linux/UNIX

- Ingres interactive tool – **iiodbcadmin**
- odbc.ini and odbc.sys.ini are located in the $II_SYSTEM/ingres/files directory

```
II_SYSTEM=/opt/Actian/IngresII
LD_LIBRARY_PATH=/opt/Actian/IngresII/ingres/lib:/opt/Actian/IngresII/ingres/lib/lp32
ODBCSYSINI=/opt/Actian/IngresII/ingres/files
export II_SYSTEM LD_LIBRARY_PATH ODBCSYSINI
```
Ingres ODBC with PHP
Setting Up ODBC Connectivity for PHP

Define the ODBC Connectivity (Connection String)

- From the options listed above:
  - DSN
  - VNODE
  - Dynamic Vnode

Install the PHP package that support the ODBC driver

- Option 1: Install the php-odbc package
  - Red Hat / CentOS
    - `yum install php-odbc`
  - Ubuntu / Debian
    - `apt-get install php5-odbc`

- Option 2: Install PDO
  - PHP Data Objects
Ingres ODBC with PHP

- Set up the user that runs the web service as an Ingres user. For example, if the user is wwwrun, adding it to the database engine should look like:
  - `~> echo "create user wwwrun;" | sql iidbdb`

- (UNIX/Linux) To check what the user that runs the web service is, you could run `ps` and `grep` for the service name:
  - E.g.: `ps -ef | grep php-fpm | grep -v grep`
  - Sample output:

```plaintext
root   11473     1  0 11:08 ?        00:00:00 php-fpm: master process (/etc/php-fpm.conf)
nginx  11474  11473  0 11:08 ?        00:00:00 php-fpm: pool www
nginx  11475  11473  0 11:08 ?        00:00:00 php-fpm: pool www
nginx  11476  11473  0 11:08 ?        00:00:00 php-fpm: pool www
nginx  11477  11473  0 11:08 ?        00:00:00 php-fpm: pool www
nginx  11478  11473  0 11:08 ?        00:00:00 php-fpm: pool www
```
Setting Up the Environment for the Web Server

Apache

- Red Hat / CentOS
  - Edit (as root) `/etc/sysconfig/httpd` to include the following

    ```
    II_SYSTEM=/opt/Actian/IngresII
    LD_LIBRARY_PATH=/opt/Actian/IngresII/ingres/lib: \
    /opt/Actian/IngresII/ingres/lib/lp32
    ODBCSYSINI=/opt/Actian/IngresII/ingres/files
    export II_SYSTEM LD_LIBRARY_PATH ODBCSYSINI
    ```

  - Create a new config file (as root), `/etc/httpd/conf.d/ingres.conf`, adding the following:

    ```
    PassEnv II_SYSTEM LD_LIBRARY_PATH ODBCSYSINI
    ```

  - Restart apache (as root). E.g.
    - Red Hat/CentOS 6.x:
      ```
      service httpd restart
      ```
    - Red Hat/CentOS 7.x:
      ```
      systemctl restart httpd.service
      ```
Setting Up the Environment for the Web Server (continued)

- Debian/Ubuntu
  - Edit /etc/apache2/envvars to include the following:

    II_SYSTEM=/opt/Actian/IngresII
    LD_LIBRARY_PATH=/opt/Actian/IngresII/ingres/lib
    ODBCSYSINI=/opt/Actian/IngresII/ingres/files
    export II_SYSTEM LD_LIBRARY_PATH ODBCSYSINI

  - Edit /etc/apache2/mods-available/ingres.conf to include the following:

    PassEnv II_SYSTEM LD_LIBRARY_PATH ODBCSYSINI

  - Activate ingres.conf:

    ln -s /etc/apache2/mods-available/ingres.conf /etc/apache2/mods-enabled/ingres.conf

  - Restart apache:

    /etc/init.d/apache2 restart
Setting Up the Environment for the Web Server (continued)

- **SUSE Linux**
  - Edit `/usr/share/apache2/load_configuration` to include the following:

    ```
    II_SYSTEM=/opt/Actian/IngresII
    LD_LIBRARY_PATH=/opt/Actian/IngresII/ingres/lib
    ODBC_SYSINI=/opt/Actian/IngresII/ingres/files
    export II_SYSTEM LD_LIBRARY_PATH ODBC_SYSINI
    ```

  - Create a new config file, `/etc/apache2/conf.d/ingres.conf`, adding the following:

    ```
    PassEnv II_SYSTEM LD_LIBRARY_PATH ODBC_SYSINI
    ```

  - Restart apache:

    ```
    /etc/init.d/apache2 restart
    ```
Setting Up the Environment for the Web Server (continued)

Setup with Nginx

- Edit the php-fpm configuration file (/etc/php-fpm.d/www.conf - or corresponding on the given OS). Add the II_SYSTEM directory value and the value of the LD_LIBRARY_PATH environment variable (or corresponding for the OS) to this file as environment parameters as shown in the example below.

  env[II_SYSTEM] = /opt/Actian/IngresII
  env[ODBCSYSINI] = /opt/Actian/IngresII/files

- Restart the php-fpm service.
Note:

- SELinux should be disabled during the install and configuration process of php-odbc with Ingres. If any problems occur after re-enabling SELinux, those are due to SELinux configuration, and not the scope of this presentation.

- With unixODBC, confirm that II_ODBC_WCHAR_SIZE is set correctly as described in Ingres ODBC Support for unixODBC (https://communities.actian.com/s/article/Ingres-ODBC-Support-for-unixODBC) – KB article.
A common error is:

PHP Warning: odbc_connect(): SQL error: [unixODBC][Driver Manager]Driver's SQLAllocHandle on SQL_HANDLE_HENV failed, SQL state IM004 in SQLConnect in /opt/www/test/php_ingres_connect/example.php on line 4
A common error is:

PHP Warning: odbc_connect(): SQL error: [unixODBC][Driver Manager]Driver's SQLAllocHandle on SQL_HANDLE_HENV failed, SQL state IM004 in SQLConnect in /opt/www/test/php_ingres_connect/example.php on line 4

Solution: II_SYSTEM needs to be set.
Examples

- Simple examples listed at
Ingres ODBC and Python
Connection Definition

• A connection string needs to be defined as described earlier

• The pyodbc module needs to be available

• In a similar way, the pypyodbc module would also work
Setting Up the Environment for Python

UNIX/Linux

In similar ways, the following must be set in the environment where the Python scripts are executed:

- \texttt{II\_SYSTEM=/opt/Actian/IngresII}
- \texttt{LD\_LIBRARY\_PATH=/opt/Actian/IngresII/ingres/lib:/opt/Actian/IngresII/ingres/lib/lp32}
- \texttt{ODBCSYSINI=/opt/Actian/IngresII/ingres/files}
- \texttt{export II\_SYSTEM LD\_LIBRARY\_PATH ODBC SYSINI}

With unixODBC, confirm that \texttt{II\_ODBC\_WCHAR\_SIZE} is set correctly as described in \textit{Ingres ODBC Support for unixODBC}.

Details and examples can be found at:
- \url{https://communities.actian.com/s/article/Ingres-ODBC-and-Python}
Ingres ODBC for Perl
Getting the Generic ODBC Driver

- DBD::ODBC is not intrinsically part of Perl, but many distributions include it by default. It is possible to build it but that will not be covered.
- Under Windows, DBD:ODBC is included in both the Active State distribution (ActivePerl, http://www.activestate.com/Products/ActivePerl/) and Strawberry Perl http://www.strawberryperl.com/.
- Under various Linux distros perl as well as DBI and DBD:ODBC is a simple apt-get, yum, yast, etc. call away. For example under Ubuntu (8.04 LTS Server edition), perl is already installed and the DBI/DBD packages can be installed with a simple:
  - sudo apt-get install libdbi-perl libdbd-odbc-perl
- Other install options are:
  - CPAN install (either using CPAN tools or simple download the tarball and build it)
Dependencies

- The DBD::ODBC module uses... ODBC, so the Ingres ODBC driver must be installed.
- Under Windows there are no further requirements.
- Under UNIX/Linux, either UnixODBC or Ingres CLI is required. As discussed above, apt-get, etc. should handle this (automatically). If you want to build DBD::ODBC you can build with Ingres CLI, if the Ingres ODBC driver has been installed CLI is already installed.
- With unixODBC, confirm that II_ODBC_WCHAR_SIZE is set correctly as described in Ingres ODBC Support for unixODBC.
Ingres ODBC for Perl

- Details and examples can be found at
  - https://communities.actian.com/s/article/Ingres-ODBC-for-Perl
Other Frameworks
Other Frameworks

- Web2Py ([www.web2py.com](http://www.web2py.com))
  - Supports Ingres along with other databases
  - Inspired by Django

- Laravel
  - PHP

- Symphony
  - PHP

- Django
  - Python

- ASP.NET
  - .NET

- There are Perl frameworks as well
OpenROAD JSON-RPC
The Geocities-izer

† Was Made By A 13 Year-Old In 1996

Type any URL in the box below and click Submit to see how it would look as a Geocities page.

Or Try one of these: The New York Times YouTube BoingBoing

http://cnn.com Submit

Some pages may work very slowly or not at all. Many webapps are just too advanced for Geocities.

Turn your sound up for the full effect.

Created by Mike Lacher

Have a problem? Want your site blocked from appearing in the Geocitiesizer? Contact us
OpenROAD

OpenROAD standard (thick) client
- (3.x, 4.0)
- Direct connectivity to the database from the client

OpenROAD Server (4.1 – 6.2)
- Initially via COM/DCOM only
- Added HTTP gatekeeper capability
- Thin clients – requiring shared libraries to be available to the web server

Set orRSO = Server.CreateObject("OpenROAD.RemoteServer")
Set orPDO = Server.CreateObject("OpenROAD.ParameterData")

JSON-RPC
OpenROAD JSON-RPC

- Traditional OpenROAD Server
  - Standard OpenROAD thin client
  - Property Changer
  - LoadNRun
OpenROAD JSON-RPC

- Traditional OpenROAD Server
  - Standard OpenROAD thin client
  - Property Changer
  - LoadNRun

- Implementation of the JSON-RPC 2.0 Specification
  - Any client supporting JSON via HTTP, including a web based client, e.g. HTML with JS, PHP, Python, Java…
OpenROAD JSON-RPC

- Standard OpenROAD thin client
- Property Changer
- LoadNRun

- Any client supporting JSON via HTTP, including a web based client, e.g. HTML with JS, PHP, Python, Java...

```javascript
$.ajax({
  type: 'POST',
  url: baseUrl,
  data: JSON.stringify({
    "jsonrpc": "2.0",
    "method": methodName,
    "id": 3,
    "params": methodParams}, null, 4),
  async: false,
  dataType: 'json',
  contentType: 'application/json',
  success: function (data) {
    // Your SUCCESS code
  },
  error: function (data) {
    isError = true;
  }
});
```
JSON-RPC Demo
Summary
How to connect Ingres to the Web?

Java & .NET

Implementations with Java – JDBC

.NET / ASP.NET

ODBC

Ingres ODBC Connectivity

- PHP
- Python *
- Perl

Other Frameworks

OpenROAD

OpenROAD JSON-RPC
Thank you!

@crisianculovici