EDT 0.8 Stakeholder Meeting

*IBM i called program support*
Stakeholder Meeting

- EDT 0.8 is currently under development
- Meeting Goal - Discuss the design and implementation of a particular feature to ensure it meets stakeholder needs
- Plans are subject to change (mostly based on your input!)
- Feel free to ask questions via the phone or chat to All.
- Press *6 to mute / unmute your phone. Please mute your phone unless you are asking a question.

- Today’s Topic: Support for IBMi
- Lead developer: Joe Vincens
What are we adding

- The ability to call IBMi programs or service programs.
  - Host program calls can only be done from a Service, Library, Program, or Handler generated to Java.
    - A function is used as the model for the host program. This is referred to as the proxy function.
    - The proxy function is invoked using a `call` statement.
- Our implementation uses the JTOpen toolkit.
The proxy function

- A Service, Library, Program, or Handler function is used as a proxy to gain access to a host program.
  - The function:
    - Body must be empty.
    - Signature must match the host program's signature
    - Must have an IBMiProgram annotation.

- IBMiProgram annotation
  ```
  Record IBMiProgram type Annotation
  programName string;
  libraryName string;
  isServiceProgram boolean = false;
  connectionResourceBindingURI String?;
  parameterAnnotations any?[];

  end
  ```
The data

- The host requires structured types but many EGL types are variable length, ie:
  - String
  - Record
  - Arrays
- Annotations control how data is converted from the EGL variable to the host structure.
## Type mapping

<table>
<thead>
<tr>
<th>AS400Type</th>
<th>EGL Type</th>
<th>EGL annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS400Array</td>
<td>EGL List</td>
<td>AS400Array*</td>
</tr>
<tr>
<td>AS400Bin1</td>
<td>smallint</td>
<td>AS400Bin1*</td>
</tr>
<tr>
<td>AS400Bin2</td>
<td>smallint</td>
<td>AS400Bin2</td>
</tr>
<tr>
<td>AS400Bin4</td>
<td>int</td>
<td>AS400Bin4</td>
</tr>
<tr>
<td>AS400Bin8</td>
<td>bigint</td>
<td>AS400Bin8</td>
</tr>
<tr>
<td>AS400ByteArray</td>
<td>bytes</td>
<td></td>
</tr>
<tr>
<td>AS400Date</td>
<td>date</td>
<td>AS400Date</td>
</tr>
<tr>
<td>AS400DecFloat</td>
<td>decimal(x,y)</td>
<td>AS400DecFloat*</td>
</tr>
<tr>
<td>AS400Float4</td>
<td>smallfloat</td>
<td>AS400Float4</td>
</tr>
<tr>
<td>AS400Float8</td>
<td>float</td>
<td>AS400Float8</td>
</tr>
<tr>
<td>AS400PackedDecimal</td>
<td>decimal(x,y)</td>
<td>AS400PackedDecimal</td>
</tr>
<tr>
<td>AS400Structure</td>
<td>Record or Handler</td>
<td>none</td>
</tr>
<tr>
<td>AS400Text</td>
<td>string</td>
<td>AS400Text*</td>
</tr>
<tr>
<td>AS400Time</td>
<td>time</td>
<td>AS400Time</td>
</tr>
<tr>
<td>AS400Timestamp</td>
<td>timestamp</td>
<td>AS400Timestamp</td>
</tr>
<tr>
<td>AS400UnsignedBin1</td>
<td>smallint</td>
<td>AS400UnsignedBin1*</td>
</tr>
<tr>
<td>AS400UnsignedBin2</td>
<td>int</td>
<td>AS400UnsignedBin2*</td>
</tr>
<tr>
<td>AS400UnsignedBin4</td>
<td>bigint</td>
<td>AS400UnsignedBin4*</td>
</tr>
<tr>
<td>AS400UnsignedBin8</td>
<td>decimal(32)</td>
<td>AS400UnsignedBin8*</td>
</tr>
<tr>
<td>AS400ZonedDecimal</td>
<td>decimal(x,y)</td>
<td>AS400ZonedDecimal*</td>
</tr>
</tbody>
</table>

* Required
Example records

record CUST
    ID int;
    CUSTNO string{AS400Text {length = 7}}; //char(7);
    CUSTNA string{AS400Text{length = 40}}; //char(40);
    REPNO string{AS400Text{length = 5}}; //char(5);
    CONTAC string{AS400Text{length = 30}}; //char(30);
    CPHONE string[]{AS400Array{elementCount = 5, elementTypeAS400Annotation = AS400Text{length = 17}};};
end

record Record1
    char20 string{AS400Text{length = 20}};
    unicode20 string{AS400Text{length = 20, encoding = "UTF-16BE"}};
    bin2sign smallint;
    bin4sign int;
    bin8sign bigint;
    dec112 decimal(11,2);
    dec15 decimal(15);
    num6 decimal(6){AS400ZonedDecimal {}};
    num103 decimal(10,3){AS400ZonedDecimal {}};
    bin2 int{AS400UnsignedBin2{}};
    bin4 bigint{AS400UnsignedBin4{}};
    date1 date;
    date2 date{AS400Date {ibmiFormat=com.ibm.as400.access.AS400Date.FORMAT_USA}};
    date3 date{AS400Date {ibmiFormat=com.ibm.as400.access.AS400Date.FORMAT_USA, ibmiSeparatorChar = null}};
    timestamp1 timestamp("yyyyMMddhhmmss");
    float4 smallfloat;
    float8 float;
end
The Proxy function

- Example:

```java
function GETCustomers(
    CUST CUST[] inout,
    //char(1)
    EOF string inout,
    COUNT decimal(2,0) inout) {
    @IBMiProgram{
        parameterAnnotations = [
            @AS400Array{elementCount = 10, returnCountVariable = COUNT},
            @AS400Text{length = 1},
            null
        ]
        programName = "GETREC"
        libraryName = "/QSYS.LIB/VARLABXX.LIB/",
        connectionResourceBindingURI = "binding:ibmi1"
    }
}
end
```
The proxy function - Generated and runtime code

The generator creates the proxy function body that:

1) Instantiates or uses an IBMiConnection.

2) An AS400DataType is created for each parameter. The AS400DataType handles the conversion from EGL data to host data (byte[]).

3) A ProgramParameter is created and populated for each parameter.

4) The ProgramCall object is created using the program name, procedure name, AS400 object, and the ProgramParameters.

5) Run the ProgramCall..

6) Ok or Exception
   - Exception
     - Throw an EGL exception wrapped around the Java exception.
   - OK
     1) Populate the EGL data using the AS400DataType. For a service program with a return convert the return value.
     2) Arrays with the AS400Array.returnCountVariable annotation field are resized using the returnCountVariable.
Resource bindings and the IBMiConnection

- Resource bindings externalize connections from your code.
  - SQL database
  - REST service
  - IBMi connections

- The IBMiConnection resource binding has:
  - Program location
    - System name
    - Library
  - Program access credentials
    - Userid
    - Password
  - Conversion Controls
    - Encoding
    - DateFormat
    - DateSeparatorChar
    - TimeFormat
    - TimeSeparatorChar
    - Timezone
IBMiConnection

- **Obtaining an IBMiConnection**
  - Get a connection from the Resource bindings.
    - The IBMiConnection fields are set to the values in the egldd entry.
      
      ```scala
      conn IBMiConnection? {Factory{}};
      
      or
      conn IBMiConnection? = SysLib.getResource("binding:conn");
      ```
    - New a connection
      
      ```scala
      conn IBMiConnection = new JTOpenConnection; //fields are null
      ```

- **After a connection is obtained, you can change any of the field values**
  
  ```java
  conn._library = "/QSYS.LIB/TSTSRVPGM.LIB";
  conn.encoding = "CP037";
  conn.password = "MYPASSWORD";
  conn.system = "AS4069.rtp.raleigh.ibm.com";
  conn.userid = "WSEDTEST"
  ```

- During the execution of the proxy function, the runtime does a IBMiConnection.getAS400 which obtains an AS400 connection from the AS400ConnectionPool.
Using the proxy function

- **Rich UI Example**
  - From a Rich UI application simply invoke the service function using the asynchronous call statement:
    ```
    CUST CUST[];
    EOF string;
    COUNT decimal(2,0);
    call TestSimpleHandler.GETCustomers(CUST, EOF, COUNT)
    returning to handlerServiceResponse onException handleException;
    ```

- From a Program, Handler, Service or Library generated to Java using a synchronous call statement.
  - **Syntax**: `call part.function([argumentList]) [using myConnection] [returns(myReturnVariable)]`
  - **Examples**
    - Using the connectionResourceBindingURI annotation field connection.
      ```
      CUST CUST[];
      EOF string;
      COUNT decimal(2,0);
      call MyHostLibrary.GETCustomers(CUST, EOF, COUNT);
      ```
    - Specifying a connection with the **using** clause
      ```
      conn IBMiConnection?(@Resource{uri="binding:ibmi1"});
      conn.password = "MYPASSWORD";
      conn.userid = "WSEDTEST";
      h1 TestSimpleHandler;
      CUST CUST[];
      EOF string;
      COUNT decimal(2,0);
      call h1.GETCustomers(CUST, EOF, COUNT) using conn;
      ```
Resources

- Specification: https://bugs.eclipse.org/bugs/show_bug.cgi?id=366706
- Search the documentation for “Accessing an IBM i called or service program”.
- JTOpen: http://jt400.sourceforge.net/
You can participate in EDT

- Share insights with the community:

  EDT wiki (http://wiki.eclipse.org/EDT)

  EDT forum (http://www.eclipse.org/forums/index.php?t=thread&frm_id=190)

  EDT blogs (http://eclipse.org/edt#community)

- Suggest enhancements and report bugs (if any):

  Bugzilla (https://bugs.eclipse.org/bugs/)

- Tell your colleagues:

  EDT project home (http://eclipse.org/edt)

- Thanks!!