Application of Java in UG Based CAD Tool Development

Xuefeng Zhang
GE Global Research
Niskayuna, NY
Outline

• Overview of Java technology
• UG based CAD tool development using Java
  — Ufunc and common API
  — Development tools
  — Execute Java application in UG
  — Debug Java application in UG
• Practical experience
  — What if Ufunc API/common API has problem
  — Load dynamic library
  — Java GUI development
  — Java thread and UG thread
  — Other practical experience
• Case study: X-section Matching tool.
Java Technology

• Java Programming Language

• Java Platform
  – Java 2 platform Standard Edition (J2SE)
  – Java 2 platform Enterprise Edition (J2EE)
  – Java 2 platform Micro Edition (J2ME)
  – Java Card technology
Java Runtime Environment

Java Source Code

Java Compiler

Byte Code

Class loader & bytecode verifier

awt  io  net  ...

Java class libraries

Interpreter

JIT

Garbage Collector

Threads and Synchronization

JVM

JVM (Intel)  Intel/Windows PC

JVM (Sun)  SPARC/Solaris Workstation

Other Architectures Supporting JVM

http://system.cmlab.csie.ntu.edu.tw/seminar/javaintr.ppt
Why Java

• Simple
• Object-Oriented
• Write once, run anywhere
• Small byte code
• High performance
• Distributed computing
• Multi-thread
• Garbage collected
• Feature rich
• Popular
• .......
• Supported by UG!
UG Based CAD Tool Development Using Java
APIs

• UG/Open for Java since NX3
• Ufunc
  — Wrapped for Java
  — Mature
  — Well documented
  — Well supported
• Common API
  — Object-Oriented
  — Journal
  — Not complete
  — Not well documented
  — Future trend
# Ufunc for Java

- Package `nxopen.uf`

<table>
<thead>
<tr>
<th>uf</th>
<th>UF</th>
<th>Contains wrappers for all 'UF' functions and data types.</th>
</tr>
</thead>
<tbody>
<tr>
<td>uf_abort</td>
<td>UFAabort</td>
<td>Contains wrappers for all 'UFAabort' functions and data types.</td>
</tr>
<tr>
<td>uf_assem</td>
<td>UFAassem</td>
<td>Contains wrappers for all 'UFAassem' functions and data types.</td>
</tr>
<tr>
<td>uf_attr</td>
<td>UFAattr</td>
<td>Contains wrappers for all 'UFAattr' functions and data types.</td>
</tr>
<tr>
<td>uf_bound</td>
<td>UFBound</td>
<td>Contains wrappers for all 'UFBound' functions and data types.</td>
</tr>
<tr>
<td>uf_brep</td>
<td>UFBrep</td>
<td>Contains wrappers for all 'UFBrep' functions and data types.</td>
</tr>
<tr>
<td>uf_cam</td>
<td>UFCam</td>
<td>Contains wrappers for all 'UFCam' functions and data types.</td>
</tr>
<tr>
<td>uf_cfi</td>
<td>UFCamBnd</td>
<td>Contains wrappers for all 'UFCamBnd' functions and data types.</td>
</tr>
<tr>
<td>uf_cgm</td>
<td>UFCamGeom</td>
<td>Contains wrappers for all 'UFCamGeom' functions and data types.</td>
</tr>
<tr>
<td>uf_clear</td>
<td>UFCamPref</td>
<td>Contains wrappers for all 'UFCamPref' functions and data types.</td>
</tr>
<tr>
<td>uf_clone</td>
<td>UFCamPrePro</td>
<td>Contains wrappers for all 'UFCamPrePro' functions and data types.</td>
</tr>
<tr>
<td>uf_csyt</td>
<td>UFCamText</td>
<td>Contains wrappers for all 'UFCamText' functions and data types.</td>
</tr>
<tr>
<td>uf_curve</td>
<td>UFCfi</td>
<td>Contains wrappers for all 'UFCfi' functions and data types.</td>
</tr>
<tr>
<td>uf_dbc_mld</td>
<td>UFCgm</td>
<td>Contains wrappers for all 'UFCgm' functions and data types.</td>
</tr>
</tbody>
</table>

```
UFPart.OpenData open(String partName)
throws NXException,
RemoteException

Refer to UF_PART_open for documentation
```
Example Code

```java
public static void main(String[] args) throws Exception {
    uf_session = (UFSession)SessionFactory.get("UFSession");

    UFPart uf_part = uf_session.part();
    UFObj uf_obj = uf_session.obj();

    UFPart.OpenData open_data = uf_part.open("test.prt");
    if(open_data.errorStatus.failed)
        throw new Exception("Failed to open the part file");

    Tag prt = open_data.part;

    String prt_name = uf_part.askPartName(prt);

    //do something

    //save the file
    uf_part.save();

    //close file
    uf_part.close(prt, 0, 1);
}
```

- Get access to ufunc APIs
- Get access to APIs defined in uf_part.h
- Same as UF_PART_open
- Return value usually saved in a data structure
- No need to call UF_free etc
## Packages

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>nxopen</code></td>
<td>Provides classes and interfaces for NX Open Common API.</td>
</tr>
<tr>
<td><code>nxopen.annotations</code></td>
<td>Provides classes and interfaces relating to annotations.</td>
</tr>
<tr>
<td><code>nxopen.assemblies</code></td>
<td>Provides classes and interfaces relating to assemblies.</td>
</tr>
<tr>
<td><code>nxopen.cae</code></td>
<td>Provides classes and interfaces relating to CAE.</td>
</tr>
<tr>
<td><code>nxopen.cam</code></td>
<td>Provides classes and interfaces relating to CAM.</td>
</tr>
<tr>
<td><code>nxopen.die</code></td>
<td>Provides classes and interfaces relating to die design.</td>
</tr>
<tr>
<td><code>nxopen.drawings</code></td>
<td>Provides classes and interfaces relating to drawings.</td>
</tr>
<tr>
<td><code>nxopen.features</code></td>
<td>Provides classes and interfaces relating to features.</td>
</tr>
<tr>
<td><code>nxopen.features.sheetmetal</code></td>
<td>Provides classes and interfaces relating to sheetmetal features.</td>
</tr>
<tr>
<td><code>nxopen.fields</code></td>
<td>Provides classes and interfaces relating to fields.</td>
</tr>
<tr>
<td><code>nxopen.geometricutilities</code></td>
<td>Provides classes and interfaces relating to geometric utilities.</td>
</tr>
<tr>
<td><code>nxopen.layer</code></td>
<td>Provides classes and interfaces relating to layers.</td>
</tr>
<tr>
<td><code>nxopen.motion</code></td>
<td>Provides classes and interfaces relating to motion.</td>
</tr>
<tr>
<td><code>nxopen.options</code></td>
<td>Provides classes and interfaces relating to options.</td>
</tr>
<tr>
<td><code>nxopen.pdm</code></td>
<td>Provides classes and interfaces relating to PDM.</td>
</tr>
<tr>
<td><code>nxopen.positioning</code></td>
<td>Provides classes and interfaces relating to positioning.</td>
</tr>
<tr>
<td><code>nxopen.preferences</code></td>
<td>Provides classes and interfaces relating to preferences.</td>
</tr>
<tr>
<td><code>nxopen.routing</code></td>
<td>Provides classes and interfaces relating to routing.</td>
</tr>
</tbody>
</table>
```
public class journal {
    public static void main(String[] args) throws NXException, java.rmi.RemoteException {
        Session theSession = (Session) SessionFactory.get("Session");
        Part workPart = theSession.parts().work();
        Part displayPart = theSession.parts().display();
        // -------------------------------
        // Menu: Insert->Design Feature->Extrude...
        // -------------------------------
        int markId1;
        markId1 = theSession.setUndoMark(nxopen.Session.MarkVisibility.VISIBLE, "Start");

        Section section1;
        section1 = workPart.sections().createSection(0.02413, 0.0254, 0.5);

        nxopen.features.Feature nullFeatures_Feature = null;
        nxopen.features.ExtrudeBuilder extrudeBuilder1;
        extrudeBuilder1 = workPart.features().createExtrudeBuilder(nullFeatures_Feature);

        extrudeBuilder1.setSection(section1);

        nxopen.geometricutilities.Limits limit1;
        limit1 = extrudeBuilder1.limits();

        nxopen.geometricutilities.FeatureOptions featureOptions1;
        featureOptions1 = extrudeBuilder1.featureOptions();

        nxopen.geometricutilities.LinearLimits linearLimits1 = ((nxopen.geometricutilities.LinearLimits) limit1);
        nxopen.geometricutilities.Extend extend1;
        extend1 = linearLimits1.startExtend();

        nxopen.geometricutilities.Extend extend2;
        extend2 = linearLimits1.endExtend();

        nxopen.geometricutilities.BooleanOperation booleanOperation1;
        booleanOperation1 = extrudeBuilder1.booleanOperation();

        Direction direction1;
```
Development Tools

- Compiler: JDK
- IDE: Eclipse, NetBeans, etc
Execute Java Application in UG

• Execute an External UG application written in Java
  – Include UG system jars in CLASSPATH
• Execute an internal UG application written in Java

• Associate toolbar button to jar/class file

```plaintext
... BUTTON DCA_cc
LABEL &DCA_cc
ACTIONS dca.jar
...

XXX.men

... BUTTON DCA_cc
LABEL &DCA_cc
BITMAP cc.bmp
...

XXX.tbr
```
Execute Java Application in UG

• Execute Java application with a different version of JRE
  – JRE coming with NX 4.0.4.2 is version 1.4.2
  – Some Java features are not available in early version
  – IR 5818247: The NXOpen jar files were inadvertently built with the wrong version of Java. Refer to development as PR 1611708

```
set UGII_JVM_LIBRARY_DIR=C:\Program Files\java\jre1.5.0_08\bin
```
Debug External UG Application with IDE

• Debug it like a normal Java application
Debug Internal UG Application with IDE

- Can not load UG from Eclipse debugger
- Debug it as a remote Java application in Eclipse
- Step 1: Launch UG with following env variable

```
set UGII_JVM_OPTIONS=-Xdebug -Xrunjdwp:transport=dt_socket,
address=8001,server=y,suspend=n
```

- 8001 is a port number. You can use any port number that is not in use.
- You can use dt_shmem instead of dt_socket if it is available for your JRE, but dt_shmem is not available on all platforms. For more information
- More details: Java Platform Debugger Architecture
Debug Internal UG Application with IDE

• Step 2: Execute any simple Java application (such as HelloWorld) from UG to activate JVM
• Step 3: Attach Eclipse Debugger to UG
Debug Internal UG Application with IDE

**Step 4. Load Java application in UG**

you can stop the debugging by using disconnect and the program will run in normal mode.
Practical Experience
What If Ufunc Doesn’t Work

• Possible reasons
  – Feature missing
  – Bug in original ufunc
  – Bug in wrapping ufunc for Java

• First choice: Use Common API to replace it

• Switch between Ufunc and Common APIs
  – Ufunc use Tag as object identifier
  – Common API use Object

• Get object tag: use method tag().
  ```java
  Part myPart = theSession.parts().newDisplay("MyPart", Part.Units.MILLIMETERS);
  theUFSession.part().askPartName(myPart.tag());
  ```

• Get object from tag: use TaggedObjectManager
  ```java
  Arc arc = (Arc) theSession.taggedObjectManager().get(arc2Tag);
  ```
Examples

- Example 1: UF_UI_select_with_class_dialog
  - Find a replacement from common API
  - Multiple selectObject and selectObjects methods in nxopen.Selection

- Example 2:

```java
/**
 * Used to replace ufunc askDatumAxisParms
 * Take object tag instead of feature tag as input
 */
public static UFModlFeatures.AskDatumAxisParmsData askDatumAxisParms(Tag objId) throws NXException, RemoteException {

UFModlFeatures.AskDatumAxisParmsData ret = new UFModlFeatures.AskDatumAxisParmsData();
DatumAxis da = (DatumAxis)DCANxUtils.nx_session().taggedObjectManager().get(objId);
Vector3d v = da.direction();
Point3d p = da.origin();
ret.normal = new double[]{v.x, v.y, v.z};
ret.origin = new double[]{p.x, p.y, p.z};
return ret;
}
```
What if Common API Doesn’t Work Neither

- If C version of Ufunc works properly, wrap ufunc with JNI (Java Native Interface) for Java by yourself

```java
static{
    System.loadLibrary("DCAHelp");
}

private static native int
AskOccsOfEntity0(int obj, int[] occs);
/**
 * This is a replacement of function
 * askOccsOfEntity in class UFAssem.
 * The one comes with UG does not work
 * properly.
 */

public static Tag[] askOccsOfEntity(Tag obj)
{
    int[] tags = new int[128];
    int n = AskOccsOfEntity0(obj.value, tags);
    if(n <= 0)
        return null;
    Tag[] ret = new Tag[n];
    for(int i=0 ; i<ret.length ; i++)
        ret[i] = new Tag(tags[i]);
    return ret;
}
```

```java
JNIEXPORT jint JNICALL
Java_dca_nx_DCANxUtils_AskOccsOfEntity0 (JNIEnv *env,
    jclass obj,
    jint obj_tag,
    jintArray occs)
{
    tag_t *ret;
    int n;
    n = 
    UF_ASSEM_ask_occocs_of_entity(obj_tag, &ret);
    (*env)->SetIntArrayRegion(env,
        occs, 0, n, ret);
    UF_free(ret);
    return n;
}
```
What if None of Above Works

• File IR
• Find a workaround
Load Dynamic Linked Library

- Reasons
  - `BaseSession.LibraryUnloadOption.IMMEDIATELY`
  - DLL get loaded by `URLClassLoader`, which ug people internally subclassed and used for loading the classes and DLL for JVM.
  - If all the classes use the native method of that DLL, they will also try to load the DLL using The new instance of `URLClassLoader`.
  - JVM imposes the restriction that DLL can be loaded only once in any class loader.

```plaintext
printf's and cout's will go here now
Fatal error: Native Library H:\aviation\dcahome\application\DCAHelp.dll already loaded in another classloader
```
Load Dynamic Linked Library

• Solution 1: avoid using option: BaseSession.LibraryUnloadOption.IMMEDIATELY
  — Classes won’t get re-loaded in second execution

```java
//public static final int getUnloadOption()
//{
//    return BaseSession.LibraryUnloadOption.IMMEDIATELY ;
//}
```

• Solution 2: Use UGII_CLASSPATH_PRELOAD
  — Classes/jars listed in UGII_CLASSPATH_PRELOAD will be loaded while JVM is created, and will not be re-loaded

```bash
set UGII_CLASSPATH_PRELOAD = CLASS_JAR_LOADING_dll
```
GUI Development Using Java

• UIStyler not for Java

• Java GUI is much more powerful and flexible than UIStyler
  – Controller
  – Layout control
  – Other

• APIs available
  – AWT (Abstract Window Toolkit)
  – Swing
  – Other: SWT (Standard Widget Toolkit), etc
public DialogDemo(JFrame frame) {
    super(new BorderLayout());
    this.frame = frame;
    customDialog = new CustomDialog(frame, "geisel", this);
    customDialog.pack();

    //Create the components.
    JPanel frequentPanel = createSimpleDialogBox();
    JPanel featurePanel = createFeatureDialogBox();
    JPanel iconPanel = createIconDialogBox();
    label = new JLabel("Click the "\"Show it!\"" button to bring up the selected dialog.",JLabel.CENTER);

    //Lay them out.
    Border padding = BorderFactory.createEmptyBorder(20,20,5,20);
    frequentPanel.setBorder(padding);
    featurePanel.setBorder(padding);
    iconPanel.setBorder(padding);
    JTabbedPane tabbedPane = new JTabbedPane();
    tabbedPane.addTab("Simple Modal Dialogs", null,
                    frequentPanel,
                    simpleDialogDesc); //tooltip text
    tabbedPane.addTab("More Dialogs", null,
                    featurePanel,
                    moreDialogDesc); //tooltip text
    tabbedPane.addTab("Dialog Icons", null,
                    iconPanel,
                    iconDesc); //tooltip text
}
Tools Enable Visual Design of GUI

- Available in many Java IDEs (Eclipse, NetBeans, JBuilder etc)
- Example: Visual Editor for Eclipse
Known Issues

• Not native to either OS or UG
• Not integrated with UG GUI. Has different behaviors.
  – Dialog is not owned by UG window
  – Dialog does not stay on top of UG window
  – Dialog does not maximize/minimize when UG window maximizes/minimizes

Solution: Method SetAlwaysOnTop since JRE 5.0
Using Standard SWING Dialogs

• Some simple UG dialog functions are not wrapped for Java
• Use Class JOptionPane.

```
JOptionPane.showMessageDialog(null, "Eggs aren’t supposed to be green", "Message", JOptionPane.INFORMATION_MESSAGE);
```

From http://java.sun.com/docs/books/tutorial/uiswing/components/dialog.html
Using Standard SWING Dialogs

- By default, the dialog created by JOptionPane is not always on top
- Invoke JOptionPane dialogs in a different way.

```java
public static void showMessageDialog(Component parentComponent, Object message, String title, int messageType)
{
    JOptionPane pane = new JOptionPane(message, messageType);
    JDialog dialog = pane.createDialog(parentComponent, title);
    dialog.setAlwaysOnTop(true);
    dialog.setModal(true);
    dialog.setVisible(true);
}
```
Java Thread and UG Thread

- JVM created by UG shares the same thread of UG
- Java code should be in a separate thread, if
  - It takes long time
  - It has its own message loop

```java
public static void main(String[] args) {
    //code ..... 
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
            try {
                //your code here
            }
            catch (Exception e) { //exception processing
            }
            finally {
                //clearnup
            }
        }
    });
}
```
Case Study: X-Section Matching Tool
X-Section Matching Tool

Parameters: Par1 = 1.0, Par2 = 230, ...  
3D models  
Cross sections

Find the parameter values to match giving x-section

Par1 = ?  
Par2 = ?  
...

Match given x-section?  
N  
Y

Optimization approach: inefficient
Methodology

1. Release constraints

2. Match

3. Measure

P0 = 1.0
P1 = 34
P4 = 6
...

\[ p3 = 21,000, p4 = 14,000, q5 = 19,000, p6 = 27,000 \]
Demo

- Parameterize a simple flange cross section
Summary

• Java technology provides a powerful, and reliable platform for UG based CAD tool development.
• Support of Java in UG is getting mature.

• Half of the development in Design Automation group of GRC is done with Java now.
Acknowledgement

• Jeff Erno (GE GRC, Niskayuna, NY)
• Ashok K. Dhakar (GE BEC, Bangalore, India)