

*Using J2EE Connector Architecture (JCA) with EP6 iView Development*

*Below is a snippet from one of hundreds of articles available to ERPtips subscribers.*

*If you would like a complimentary copy of the full article, please email*

**[Mark.Downs@ERPtips.com](mailto:Mark.Downs@ERPtips.com)**

*(include the title of the article in your email)*

*To subscribe to ERPtips, go to*

**[www.ERPtips.com/Subscribe.asp](http://www.ERPtips.com/Subscribe.asp)**

*ERPtips Journal is published by Klee Associates, Inc.*

*ERPtips University provides both public and onsite training for SAP clients.*

*For more about ERPtips University, including the current schedule, click here:*

**[www.ERPtips.com/WorkshopSchedule.asp](http://www.ERPtips.com/WorkshopSchedule.asp)**

---

*Using J2EE Connector Architecture (JCA) with EP6 iView Development*

Bill explains how the latest version of Enterprise Portals (EP6) uses the Connector Framework instead of the SAP Java Connector. Therefore, upgrading from EP5 to EP6 can be an adventure. SAP has documentation on the subject, but sorting through it can be a task. Bill has already encountered and resolved a number of issues with SAP's J2EE Connector Architecture (JCA) that weren't addressed in SAP's own documentation. So before proceeding with EP6 projects, you'll want to make sure your development team reads this article and learns about how Bill located the missing jar files and resolved the "no class definition found" error.

Click here to read this Snippet

## Conclusions

The JCA functionality provided by SAP, used to access an SAP system, is a big improvement over the previous Java Connector and JCO-ClientService packages. There are a number of pitfalls associated with the implementation, which I have covered in this article, but these can be overcome with a little patience and are not show stoppers. Future releases should clear up some of the problems. Once the framework is up and running, it works well, provides very good response time, and yields a reliable and stable result.

In future articles, I will cover more details concerning JCA, hopefully covering the intricacies of using JDBC with JCA as a means to access a database directly from the Portal. From here, we can hopefully dive into some concepts with Enterprise Java Beans and the Portal, once these features become supported in future releases.

**Bill Guderian**, *BK Systems, Inc.* Bill is an independent contractor specializing in SAP Enterprise Portals and similar developments involving the Java programming language. He has almost ten years of SAP experience in a wide range of technical and functional areas, including extensive experience as an ABAP developer. Bill has spent the past four years focusing on the space where the SAP applications intersect with the Java programming language. He is a consultant in helping companies develop SAP bolt-on applications written in Java. Bill's email address is [Bill.Guderian@SAPtips.com](mailto:Bill.Guderian@SAPtips.com).

```
public class JCAFunction implements Function {
    /* (non-Javadoc)
     * @see sapaccess.Function#execute(com.sapportals.connector.connection.IConnection)
     */
    IInteraction interaction = null;
    IInteractionSpec specification = null;
    MappedRecord input = null;
    RecordFactory recordFactory = null;
    String bapi = null;
    IConnection connection = null;

    public JCAFunction(String bapi) throws ResourceException {
        this.bapi = bapi;
    }

    public synchronized MappedRecord execute()
        throws ConnectorException, ExecutionException {
        MappedRecord output =
            (MappedRecord) interaction.execute(specification, input);
        return output;
    }

    /* (non-Javadoc)
     * @see sapaccess.Function#setParameter(java.lang.String, java.lang.Object)
     */
    public synchronized void setParameter(String name, Object value) {
        input.put(name, value);
    }
}
```

Figure 8: Part 1 of JCAFunction Class

```
public synchronized void initialize(IConnection connection)
    throws ResourceException {
    this.interaction = connection.createInteractionEx();
    this.specification = interaction.getInteractionSpec();
    specification.setPropertyValue("Name", bapi);
    this.recordFactory = interaction.getRecordFactory();
    this.input = recordFactory.createMappedRecord("input");
}

/* (non-Javadoc)
 * @see sapaccess.Function#getBapiName()
 */
```

Figure 9: Part 2 of JCAFunction Class

```
try {
    recordSet.beforeFirst();
    while (recordSet.next()) {
        row = new Vector();
        row.add(recordSet.getString(0));
        table.addElement(row);
    }
} catch (ConnectorException e) {
}

return table;
```

Figure 10: Example of IRecordSet Usage

# SAPtips *Journal*

*The information in our publications and on our Website is the copyrighted work of Klee Associates, Inc. and is owned by Klee Associates, Inc. NO WARRANTY: This documentation is delivered as is, and Klee Associates, Inc. makes no warranty as to its accuracy or use. Any use of this documentation is at the risk of the user. Although we make every good faith effort to ensure accuracy, this document may include technical or other inaccuracies or typographical errors. Klee Associates, Inc. reserves the right to make changes without prior notice. NO AFFILIATION: Klee Associates, Inc. and this publication are not affiliated with or endorsed by SAP AG, SAP AG software referenced on this site is furnished under license agreements between SAP AG and its customers and can be used only within the terms of such agreements. SAP AG and mySAP are registered trademarks of SAP AG. All other product names used herein are trademarks or registered trademarks of their respective owners.*