How To Integrate xMTT and xMAM

Version <1.00> - 10 2007

Applicable Releases:
xMTT 2.0
xMAM 3.0
1 Scenario
You want to have some integration between the xApps Mobile Time and Travel (xMTT) and xApps Mobile Asset Management (xMAM) applications on the client.

2 Introduction
This document explains how you can create an integration scenario between xMTT and xMAM on the client.

This is achieved by exchanging offline data between both applications on the client and then displaying this data as required in the applications' user interface.

The three concepts that are used to implement an integration scenario are:
- Data exchange using Mobile Infrastructure Services API – MultiObjectFileStorage
- xMAM Customization Links
- Customer Enhancement in xMAM and xMTT to store, access, and display common data

The scope of this document is limited to the exchange of data on the client side of the integration between xMTT and xMAM. The logic behind the synchronization behavior of the two applications to the backend is based upon the customer’s individual business rules and can be handled by custom implementations.
3 Integration Implementation Concepts

3.1 Data Exchange using *Mobile Infrastructure Services API – MultiObjectFileStorage*

To allow different mobile applications to exchange data, the *Mobile Infrastructure Services API*, *MultiObjectFileStorage*, can be used.

The *MultiObjectFileStorage* class provides a persistence service based on object serialization to files. This class stores serializable objects in the file system with each object tree being stored in a separate file.

It allows applications to store objects in a common area. These objects can then be accessed from within different mobile application to meet specific requirements such as integration.

💡 The logic on when to write and read common data from a mobile application needs to be handled by the custom implementation and is based on the required business scenarios.

Customers have the flexibility to do so at any point in the application by extending the standard mobile applications accordingly using the *Mobile Client Application Framework (mCAF)*.
3.2  xMAM Customization Links

The standard xMAM application allows you to easily customize the values for xMTT links. These links get automatically created and displayed in the xMAM application in the Order Detail screen and in the Order Operation Detail screen.

If customers want to have a custom link to the xMTT application elsewhere in the xMAM application, they can do so by implementing a customer enhancement to xMAM.

The links are specified in the Define Scenario section of the SAP xMAM customization settings using the SAP Reference IMG.

Choose SAP Customizing Implementation Guide → Plant Maintenance and Customer Service → Maintenance and Service Processing → Mobile Asset Management → Define Scenario

In the Define Scenario section, you need to set the following indicators and values for the respective Business Processes for the individual SAP xMAM scenarios:

- Time Rec. Active
- Travel Exp. Active
- Time URL
- Travel URL

These settings allow you to activate and set the different URL values for the links to the xMTT Time Sheet and Travel Expenses application.

3.2.1 Customization Links Parameters

The xMAM customization links (Order Detail screen or Order Operation Detail screen) send the following key-value pairs to the context in order to be captured by the xMTT enhancement logic for further processing:

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAUFNR</td>
<td>Work Order</td>
<td>000000082805</td>
</tr>
<tr>
<td>VORNR</td>
<td>Activity/Operation</td>
<td>0020</td>
</tr>
<tr>
<td>UVORN</td>
<td>Sub-Activity/Sub-Operation</td>
<td>0010</td>
</tr>
<tr>
<td>CLIENTURL</td>
<td>xMAM Back URL Value</td>
<td>/XMAM30_LAPTOP_SR04_RELEASED/order/OrderDetail.do (Order Detail screen)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/XMAM30_LAPTOP_SR04_RELEASED/order/OperationDetail.do (Order Operation Detail screen)</td>
</tr>
<tr>
<td>CLIENTURLTEXT</td>
<td>xMAM Back URL Text</td>
<td>Back to xMAM</td>
</tr>
</tbody>
</table>

If customers want to have different parameters sent to the context, they can do so by implementing a customer enhancement to xMAM.

CLIENTURL and CLIENTTEXT are only supported as of xMAM 3.0 SP5.
Please refer to ZStartupController for sample code on how to retrieve this information from the context.

3.3 Customer Enhancement in xMAM and xMTT

Logic to store, access, and display work orders in xMAM and xMTT is built as a customer enhancement to the standard applications using the Mobile Client Application Framework (mCAF).

In order to enhance a mobile application you need to keep the standard coding as is and extend the application to make the application behave in a different way.

The trick to do this is to extend the standard classes and standard view files and configure the application in a different way so that the enhanced classes are used instead of the standard classes.

You can find more detailed information about the mCAF process in the xApps Mobile Asset Management 3.0 Enhancement Guide and xApps Mobile Time and Travel 2.0 Enhancement Guide available at http://service.sap.com/instguides.

The integration scenario described in this document is a sample scenario. Customers can adapt the scenario to suit their specific needs.

Please refer the section Step-by-Step Solution for detailed steps on implementing the sample integration scenario.

4 Integration Scenario Process Flows

The following describes the process flows that are handled by the sample integration between xMAM and xMTT.

4.1 Flow from within a specific Mobile Asset Management Work Order

1. Maintain work order from Mobile Asset Management
You perform an activity on a work order and want to enter time against that order. You want the time to be synchronized to the SAP Time Sheet component in the SAP backend system.

2. Launch Mobile Time Sheet from Mobile Asset Management
You launch the Mobile Time Sheet application from the link in the Mobile Asset Management Order Management – Order Detail Screen or Operation Detail Screen.

3. System Pre-Populates Work Order Data into Mobile Time Sheet
When you arrive in the Time Sheet Grid, work order, activity/operation, and/or sub-activity/sub-operation data are pre-populated in the Time Sheet Grid.

4. Create/Update Time Information for Work Orders in Mobile Time Sheet
You enter/update time for work orders.

5. Create/Update Time Information for non Work Orders in Mobile Time Sheet
You enter/update time for non work orders.

6. Synchronize Mobile Time Sheet data
When you choose Synchronize, the time information data is synchronized to SAP Time Sheet component in the SAP backend system.
4.2 Flow from Mobile Time Sheet

1. **Launch Mobile Time Sheet**
   You launch the Mobile Time Sheet application from the link in the Mobile Infrastructure Home Page.

2. **System Pre-Populates Search Help Work Order in Mobile Time Sheet**
   When you arrive in the Time Sheet Grid, the work order search help has an icon displayed.

3. **List Mobile Asset Management Work Orders in Mobile Time Sheet**
   When you select work order search help icon, you can select one from a list of Mobile Asset Management work orders.

4. **Create/Update Time Information for Work Orders in Mobile Time Sheet**
   You enter/update time for work orders.

5. **Create/Update Time Information for non Work Orders in Mobile Time Sheet**
   You enter/update time for non work orders.

6. **Synchronize Mobile Time Sheet data**
   When you choose Synchronize, the time information data is synchronized to SAP Time Sheet component in the SAP backend system.
5 Enhanced \textit{xMAM} and \textit{xMTT} Versions

5.1 Flow from within a specific \textit{Mobile Asset Management} Work Order

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{Mobile Asset Management Order Detail Screen}
\end{figure}
Figure 2 - Mobile Time Sheet Home Screen
Figure 3 - Mobile Time Sheet Grid with Pre-Populated Work Order, Activity/Operation, and/or Sub-Activity/Sub-Operation
5.2 Flow from Mobile Time Sheet

Figure 4 - Mobile Infrastructure Home Page
Figure 5 - Mobile Time Sheet Home Page

<table>
<thead>
<tr>
<th>Working Time</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Record Working Time</td>
<td>Record your working times here. You have recorded adequate times for all workdays until 1/1/2000. Your recorded times have been approved until 1/1/2000.</td>
</tr>
<tr>
<td>Record Working Time for Other Employees</td>
<td>Here, you can record working times for other employees.</td>
</tr>
<tr>
<td>Release Working Time</td>
<td>Release your working times to your manager. He or she will review and approve them.</td>
</tr>
<tr>
<td>Release Working Times for Other Employee</td>
<td>You can release working time of other employees to their manager.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Synchronization</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronize Data</td>
<td>Your last synchronization took place on 1/1/2007 at 10:35.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset Time Sheet</td>
<td>Initialize your database. This will remove all time-related data from your device.</td>
</tr>
<tr>
<td>Report Time Sheets for Other Employees</td>
<td>Initialize the database for other employees; this will remove all time-related data from the device.</td>
</tr>
</tbody>
</table>
Figure 6 - Mobile Time Sheet Grid with Receiver Order Search Help Icon
Figure 7 - Mobile Time Sheet Receiver Order Search Help Screen
Figure 8 - Mobile Time Sheet Receiver Order Search Help Screen Selection
6 Step-by-Step Solution

6.1 Mobile Infrastructure Services API – MultiObjectFileStorage

In this scenario, the main objects stored in the MultiObjectFileStorage API for display in xMTT are xMAM orders.

In this scenario, the MultiObjectFileStorage API is called from various points. Please refer to Customer Enhancement Sequence Diagram for details.
6.2 xMAM Customization Links Settings

In the sample scenario, we have chosen to have the Mobile Time Sheet customization links in xMAM go to the Mobile Time Sheet Home Page.

In our sample scenario, the Time URL value is “/MTS/start”. “MTS” is the name of the mobile component for Mobile Time Sheet.

<table>
<thead>
<tr>
<th>Time and Travel Notebook Integration</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Rec. Active</td>
<td>✓</td>
</tr>
<tr>
<td>Travel Exp. Active</td>
<td>✓</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time URL</th>
<th>/MTS/start</th>
</tr>
</thead>
<tbody>
<tr>
<td>Travel URL</td>
<td>/MTR/start</td>
</tr>
</tbody>
</table>

6.3 xMAM Enhancement Implementation

This section will show you how to enhance the Mobile Asset Management application to create the sample integration logic.

The steps will be to create a Mobile Asset Management development project, implement the integration logic, and export your project into a new MAM.war file which would include the new integration logic.

You can find more detailed information about this process in the xApps Mobile Asset Management 3.0 Enhancement Guide available at http://service.sap.com/instguides.

6.3.1 Prerequisites

Set up the Mobile Asset Management project in SAP NetWeaver Developer Studio (NWDS) as described in the “Development Cycle” chapter of the xApps Mobile Asset Management 3.0 Enhancement Guide available at http://service.sap.com/instguides.

6.3.2 Procedure

In our case, we will be extending the standard HomeMain.view, HomeMain.jsp, and HomeMain.java files.

You can find more information about this process in the xApps Mobile Asset Management 3.0 Enhancement Guide available at http://service.sap.com/instguides.
Please refer to the Appendix section for more details on the implementation logic.
6.3.3 Sequence Diagram

Figure 10 - Sequence Diagram Flow from MAM
6.3.4 Step-By-Step Procedure

1. Create the helper Java classes.

   Right-click on <MAM NWDS Project Root>/java, select “New -> Class”

   Enter “zcom.sap.mbs.common.integration” for the package name and “ChildObject” for the class name.

   Click on Finish.

   Copy and paste the code for this Java class available in the Appendix. Save your changes.

2. Repeat previous step for helper Java classes:
   - Constants
   - ObjectField
   - ParentObject
   - RootObject
3. Create the `ZHomeManagement` Java class.

Right-click on `<MAM NWDS Project Root>/java`, select “New -> Class”

Enter “zcom.sap.mbs.mam.home.control” for the package name, “ZHomeManagement” for the class name, and “com.sap.mbs.mam.home.control.HomeManagement” for the Superclass name.

Click on Finish.

Copy and paste the code for this Java class available in the Appendix. Save your changes.
4. Create the `ZHomeMain.view` file.

Right-click on `<MAM NWDS Project Root>/app-root/WEB-INF/home`, select "New -> File".

Enter "ZHomeMain.view" for the File name.

Click on Finish.

Copy and paste the code for this file available in the Appendix. Save your changes.
5. Create the `ZHomeMain.jsp` file.

Right-click on `<MAM NWDS Project Root>/app-root/home`, select "New -> File".

Enter "ZHomeMain.jsp" for the File name.

Click on Finish.

Copy and paste the code for this file available in the Appendix. Save your changes.
6. Create the `view.configure` file.

   Right-click on `<MAM NWDS Project Root>/app-root/WEB-INF`, select “New -> File”.

   Enter “`view.configure`” for the File name.

   Click on Finish.

   Copy and paste the code for this file available in the Appendix. Save your changes.

7. Make sure to save your updated files, export your project, and test your enhancement logic.
6.4 xMTT Enhancement Implementation

This section will show you how to enhance the Mobile Time Sheet application to create the sample integration logic.

The steps will be to create a Mobile Time Sheet development project, implement the integration logic, and export your project into a new MTS.war file which would include the new integration logic.

You can find more detailed information about this process in the xApps Mobile Time Sheet 2.0 Enhancement Guide available at http://service.sap.com/instguides.

6.4.1 Prerequisites

Set up the Mobile Time Sheet project in SAP NetWeaver Developer Studio (NWDS) as described in the “Development Cycle” chapter of the xApps Mobile Time Sheet 2.0 Enhancement Guide available at http://service.sap.com/instguides.

6.4.2 Procedure

In our case, we will be extending the standard startup.view, StartupController.java, GridAdapterImpl.java and PicklistProcessImpl.java files.

You can find more information about this process in the xApps Mobile Time Sheet 2.0 Enhancement Guide available at http://service.sap.com/instguides.
Please refer to the Appendix section for more details on the implementation logic.
6.4.3 Sequence Diagram

Figure 11 - Sequence Diagram Flow from MTS
6.4.4 Step-By-Step Procedure

1. Create the helper Java classes.

   Right-click on `<MTS NWDS Project Root>/java`, select “New -> Class”

   Enter “zcom.sap.mbs.common.integration” for the package name and “ChildObject” for the class name.

   Click on Finish.

   Copy and paste the code for this Java class available in the Appendix. Save your changes.

2. Repeat previous step for helper Java classes:
   - Constants
   - ObjectField
   - ParentObject
   - RootObject
3. Create the `ZStartupController` Java class.

Right-click on `<MTS NWDS Project Root>/java`, select “New -> Class”

Enter “zcom.sap.mbs.mts.home.control.impl” for the package name, “ZStartupController” for the class name, and “com.sap.mbs.mts.home.control.impl.ZStartupController” for the Superclass name.

Click on Finish.

Copy and paste the code for this Java class available in the Appendix. Save your changes.

4. Create the `ZGridAdapterImpl` Java class.

Right-click on `<MTS NWDS Project Root>/java`, select “New -> Class”

Enter “zcom.sap.mbs.mts.model.grid.impl” for the package name, “ZGridAdapterImpl” for the class name, and “com.sap.mbs.mts.model.grid.impl.ZGridAdapterImpl” for the Superclass name.

Click on Finish.

Copy and paste the code for this Java class available in the Appendix. Save your changes.
5. Create the `ZPicklistProcessImpl` Java class.

Right-click on `<MTS NWDS Project Root>/java`, select “New -> Class”


Click on Finish.

Copy and paste the code for this Java class available in the Appendix. Save your changes.
6. Create the zstartup.jsp file.

Right-click on `<MTS NWDS Project Root>/app-root/home`, select “New -> File”.

Enter “zstartup.jsp” for the File name.

Click on Finish.

Copy and paste the code for this file available in the Appendix. Save your changes.
7. Create the `view.configure` file.

Right-click on `<MTS NWDS Project Root>/app-root/WEB-INF`, select "New -> File".

Enter "view.configure" for the File name.

Click on Finish.

Copy and paste the code for this file available in the Appendix. Save your changes.
8. Create the zstartup.view file.

Right-click on <MTS NWDS Project Root>/app-root/WEB-INF/home, select “New -> File”.

Enter "view.configure" for the File name.

Click on Finish.

Copy and paste the code for this file available in the Appendix. Save your changes.
9. Create the `view.configure` file.

    Right-click on `<MTS NWDS Project Root>/app-root/WEB-INF`, select
    “New -> File”.

    Enter “view.configure” for the File name.

    Click on Finish.

    Copy and paste the code for this file available in the Appendix. Save your
    changes.
10. Create the `zcore.configure` file.

Right-click on `<MTS NWDS Project Root>/app-root/WEB-INF`, select "New -> File".

Enter "zcore.configure" for the File name.

Click on Finish.

Copy and paste the code for this file available in the Appendix. Save your changes.
7 Appendix

Below is the source code for the enhancement files for *xMAM* and *xMTT*.

7.1 *xMAM* and *xMTT* Helper Classes

These classes were created to have common objects used in *xMTT* and *xMAM* for the data storage. They are included in both applications.

Alternatively, customers can use any Java APIs (Array, Vectors, etc) as long as they are serializable objects.

7.1.1 ChildObject

```java
package zcom.sap.mbs.common.integration;

import zcom.sap.mbs.common.integration.ChildObject;
import zcom.sap.mbs.common.integration.RootObject;

public class ChildObject extends RootObject {

    private RootObject parent;

    public RootObject getParent() {
        return parent;
    }

    public void setParent(RootObject parent) {
        this.parent = parent;
    }

    public String toString() {
        StringBuffer result = new StringBuffer(" 
	{ Child Level ").append(getChildLevel()).append("(parent ID = ").append(parent.getId()).append(")
	{ Child Level ").append(printDetails()).append(" }
	{ Child Level ").append(printChildren());
        return result.toString();
    }

    /**
     * Returns the level of this instance in the Parent/Child tree.
     * @return
    */
```
protected int getChildLevel(){
    int result = 1;
    RootObject parent = getParent();
    while (parent instanceof ChildObject){
        result++;
        parent = ((ChildObject)parent).getParent();
    }
    return result;
}

7.1.2 Constants

package zcom.sap.mbs.common.integration;
public class Constants {
    public static final String MOFS_KEY_MAM_DATA = "MAM_DATA";
    public static final String MOFS_KEY_MAM_CALLED_FROM_MTS = "MAM_CALLED_FROM_MTS";
    public static final String CONTEXT_KEY_CALL_MTS = "CALL_MTS";
    public static final String VALUE_FALSE = "false";
    public static final String VALUE_TRUE = "true";
}

7.1.3 ObjectField

package zcom.sap.mbs.common.integration;

import java.io.Serializable;

/**
 * Represents a field in the ParentObject or ChildObject classes.<br>
 * Stores the name of the field along with its value.
 * *
 */
public class ObjectField implements Serializable {
    private String name;
    private String value;

    public ObjectField(String name, String value){
        this.name = name;
        this.value = value;
    }
public ObjectField(String name) {
    this.name = name;
}

public String getName() {
    return name;
}

public String getValue() {
    return value;
}

public void setValue(String value) {
    this.value = value;
}

public String toString() {
    return new StringBuffer("[name = ").append(name).append(", value = ").append(value).append("]").toString();
}
}

7.1.4 ParentObject

package zcom.sap.mbs.common.integration;

import zcom.sap.mbs.common.integration.ParentObject;

public class ParentObject extends RootObject {

    public String toString() {
        StringBuffer result = new StringBuffer("PARENT OBJECT { ");
        result.append(printDetails()).append(printChildren()).append("} ");
        return result.toString();
    }
}

7.1.5 RootObject

package zcom.sap.mbs.common.integration;

import java.io.Serializable;
import java.util.ArrayList;
public abstract class RootObject implements Serializable{
    protected Map children = new HashMap();
    protected Map fields = new HashMap();

    private String name;
    private String id;

    public String getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public void setId(String id) {
        this.id = id;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void addChild(ChildObject child) {
        child.setParent(this);
        children.put(child.getId(), child);
    }

    public void addChildren(List children) {
        if (children != null) {
            Iterator it = children.iterator();
            while (it.hasNext()) {
                Object o = it.next();
                if (o instanceof ChildObject == false) {
                    throw new RuntimeException("Instances must implement ChildObject interface");
                }
                addChild((ChildObject) it.next());
            }
        }
    }

    public void addChild(ChildObject child) {
        child.setParent(this);
        children.put(child.getId(), child);
    }

    public void addChildren(List children) {
        if (children != null) {
            Iterator it = children.iterator();
            while (it.hasNext()) {
                Object o = it.next();
                if (o instanceof ChildObject == false) {
                    throw new RuntimeException("Instances must implement ChildObject interface");
                }
                addChild((ChildObject) it.next());
            }
        }
    }
}
public boolean containsChildren() {
    return children != null && children.size() > 0;
}

public ChildObject getChildById(String id) {
    return (ChildObject)children.get(id);
}

public List getChildren() {
    return new ArrayList(children.values());
}

public List getChildrenByName(String name) {
    List result = new ArrayList();
    if (name == null) {
        return result;
    }
    Iterator it = children.values().iterator();
    while (it.hasNext()) {
        RootObject obj = (RootObject)it.next();
        if (name.equals(obj.getName())) {
            result.add(obj);
        }
        result.addAll(obj.getChildrenByName(name));
    }
    return result;
}

public ObjectField getField(String name) {
    return (ObjectField) fields.get(name);
}

public List getFields() {
    return new ArrayList(fields.values());
}

public String[] getFieldNames() {
    return (String[]) fields.keySet().toArray(new String[fields.size()]);
}
public void addField(ObjectField field) {
    if (field != null) {
        fields.put(field.getName(), field);
    }
}

protected StringBuffer printDetails() {
    StringBuffer result = new StringBuffer();
    result.append("[id = ").append(id).append("], name = ").append(name).append("]");
    result.append(" Fields: ");
    Iterator it = fields.values().iterator();
    while (it.hasNext()) {
        result.append(it.next().toString());
    }
    return result;
}

protected StringBuffer printChildren() {
    StringBuffer result = new StringBuffer();
    Iterator it = children.values().iterator();
    while (it.hasNext()) {
        result.append(it.next().toString());
    }
    return result;
}

7.2  xMAM Enhancement Implementation Source Code

7.2.1  ZHomeManagement

package zcom.sap.mbs.mam.home.control;

import java.util.ArrayList;
import java.util.List;
import com.sap.ip.me.api.services.MultiObjectFileStorage;
import com.sap.ip.me.api.user.UserManager;
import com.sap.mbs.core.api.BOException;
import com.sap.mbs.core.api.BOList;
import com.sap.mbs.core.api.Forward;
import com.sap.mbs.core.api.Forwards;
import com.sap.mbs.mam.bo.Mam001;
import com.sap.mbs.mam.bo.Mam001Item080;
import com.sap.mbs.mam.bo.Mam001Manager;
import com.sap.mbs.mam.home.control.HomeManagement;
import zcom.sap.mbs.common.integration.ChildObject;
import zcom.sap.mbs.common.integration.ParentObject;
import zcom.sap.mbs.common.integration.ObjectField;
import zcom.sap.mbs.common.integration.Constants;

public class ZHomeManagement extends HomeManagement {
    public final Mam001Manager orderManager;

    public ZHomeManagement() {
        orderManager =
        (Mam001Manager) naming.lookup(
            com.sap.mbs.mam.bo.Mam001Manager.class.getName());
    }

    public Forward onLoad(Forwards forwards) {
        System.out.println("in " + getClass().getName());
        MultiObjectFileStorage mofs =
        MultiObjectFileStorage.createInstance(UserManager.getInstance().getCurrentUser().getUniqueID());
        // Persist latest MAM Orders in MI API MultiObjectFileStorage for use in MTT
        mofs.write(Constants.MOFS_KEY_MAM_DATA,createData());
        // Check if MAM was called from MTT to update MAM data and forward back to MTT
        String calledFromMTS = (String)
        mofs.read(Constants.MOFS_KEY_MAM_CALLED_FROM_MTS);
        if (calledFromMTS != null && calledFromMTS.equals(Constants.VALUE_TRUE)) {
            // Reset value in MultiObjectFileStorage to false
            mofs.write(Constants.MOFS_KEY_MAM_CALLED_FROM_MTS, Constants.VALUE_FALSE);
            // Set value in context for automatic forward to MTT
            getContext().setValue(Constants.CONTEXT_KEY_CALL_MTS, Constants.VALUE_TRUE);
        }
        return super.onLoad(forwards);
    }

    private List createData() {
        List result = new ArrayList();
        BOList orders = null;
        try {
            orders = orderManager.getMam001s();
        } catch (Exception e) {
            e.printStackTrace(); // Handle exception
        }
    }
}
try {
    for (int i = 0, maxi = orders.size(); i < maxi; i++) {
        Mam001 currentOrder = (Mam001) orders.get(i);
        BOList operations = orderManager.getMam001Item080s(currentOrder);
        String orderID = (String) currentOrder.getOrderid();
        String orderShortText = (String) currentOrder.getShortText();

        //Create Order Parent Object
        ParentObject order = new ParentObject();
        order.setId(orderID);
        order.setName("order");
        ObjectField fieldOrder = new ObjectField("description", orderShortText);
        order.addField(fieldOrder);

        //Create Order Children
        for (int ops = 0, maxOps = operations.size(); ops < maxOps; ops++) {
            Mam001Item080 currentOperation = (Mam001Item080) operations.get(ops);
            String activityId = (String) currentOperation.getActivity();
            String activityDescription = (String) currentOperation.getDescription();

            ChildObject activity = new ChildObject();
            activity.setId(activityId);
            activity.setName("activity");
            ObjectField fieldActivity = new ObjectField("description", activityDescription);
            activity.addField(fieldActivity);

            String subActivityId = (String) currentOperation.getSubActivity();
            if (!subActivityId.equals("")) {
                ChildObject subActivity = new ChildObject();
                subActivity.setId(subActivityId);
                subActivity.setName("subActivity");
                activity.addChild(subActivity);
            }
            order.addChild(activity);
        }
        result.add(order);
    }
} catch (BOException e1) {
7.2.2  ZHomeMain.view

```xml
<?xml version="1.0" encoding="UTF-8"?>

<view screen="/home/ZHomeMain.jsp"
    controller="zcom.sap.mbs.mam.home.control.ZHomeManagement"
    parent_view="HomeMain">
</view>
```

7.2.3  ZHomeMain.jsp

```jsp
<% com.sap.mbs.core.api.Context ct2 = (com.sap.mbs.core.api.Context)request.getAttribute("ct");
    String callMTS =
        (String)ct2.getValue(zcom.sap.mbs.common.integration.Constants.CONTEXT_KEY_CALL_MTS);
    if (callMTS != null &&
        callMTS.equals(zcom.sap.mbs.common.integration.Constants.VALUE_TRUE)){%
<HTML>
<body onload="javascript:document.location.replace('/
    MTS
/start');">
</body>
</HTML>%>

<% } else {%

<!--This JSP was generated on Fri Apr 27 06:22:51 CEST 2007-->
<%@ include file="/include/common_jsp.inc" %>
<%@ include file="/include/common_page.inc" %>
<%@ include file="/include/common_error_message.inc" %>
<%
    String layoutCount = "0";
    TrayHelper th = TrayHelper.getInstance();
    %>
```
ViewSwitchHelper vsHelper = ViewSwitchHelper.getInstance();
ArrayList traysId = new ArrayList();

<ui:page is508="true" RTL="false" margin="false" title="Mobile Assets Management"
locale="$CT.Locale" dateFormat="<%=helper.getDateFormat()%>" onload="mainPageOnLoad()">
<inc:include file="/include/common_popup.jsp"></inc:include>
<inc:include file="/include/topHeader.jsp"></inc:include>
<%Layout layout = new Layout();
Container co1 = new Container("HomeMain","HOME_TAB_TOP_MAM_WELCOME","none");
layout.addContainer(co1);
co1.setCollapsed(false);
Content ct1co1 = new Content(co1,"HomeMain",true,false);
co1.addContent(ct1co1);
%
<ui:inputfield id="saveStateFrameId" labelText="" tooltip="" hidden="true"
value="<%=layout.getSaveStateFrameId()%>"></ui:inputfield>
<%
if (show_err) {
show_err = false;
<%> <ui:messagebar id="msgbar_HomeMain_text" text="" error_mess ">
</ui:messagebar>
<%}%>
<ui:gridlayout id="gridlayout_HomeMain" width="100%" shifting="true">
<ui:gridlayoutrow paddingTop="5px" paddingBottom="10px">
<ui:gridlayoutcell id="gridlayoutcell_HomeMain" colSpan="1" rowSpan="1" HAlign="LEFT">
<inc:include file="/home/home_layout.jsp"></inc:include>
</ui:gridlayoutcell>
</ui:gridlayoutrow>
<%String appendedTraysId =helper.convertTraysIds(traysId);%>
<ui:inputfield id="traysID" labelText="" tooltip="" hidden="true"
value="<%=appendedTraysId%>"></ui:inputfield>
<ui:inputfield id="layoutName" labelText="" tooltip="" hidden="true" value="HomeMain"></ui:inputfield>
Where “MTS” is the name of the Mobile Time Sheet Mobile Application Component deployed locally.

### 7.2.4 view.configure

```java
home.HomeMain=home.zHomeMain
```

### 7.3 xMTT Enhancement Implementation Source Code

#### 7.3.1 ZStartupController

```java
package zcom.sap.mbs.mts.home.control.impl;

import zcom.sap.mbs.common.integration.Constants;
import com.sap.mbs.core.api.Forward;
import com.sap.mbs.core.api.Forwards;
import com.sap.mbs.mts.common.structure.ICatsField;
import com.sap.mbs.mts.home.control.impl.StartupController;
import com.sap.mbs.mts.model.grid.GridAdapter;
import com.sap.ip.me.api.services.MultiObjectFileStorage;
import com.sap.ip.me.api.user.UserManager;

public class ZStartupController extends StartupController {
    public static final String FORWARD_CALL_MAM = "callMAM";
    public static String activeOrder = null;
    public static String activeActivity = null;
    public static String activeSubActivity = null;

    public ZStartupController() {
    }

    public Forward onLoad(Forwards f) {
        MultiObjectFileStorage mofs =
                MultiObjectFileStorage.createInstance(UserManager.getInstance().getCurrentUser().getUniqueID());
        System.out.println("in " + getClass().getName());
```
String orderID = (String) getContext().getValue(ICatsField.RAUFNR);
String activity = (String) getContext().getValue(ICatsField.VORNR);
String subActivity = (String) getContext().getValue(ICatsField.UVORN);

// If customers would like to return to MAM, a return URL value and text is available below
String clientURL = (String) getContext().getValue("CLIENTURL");
String clientURLText = (String) getContext().getValue("CLIENTURLTEXT");

if (orderID != null) {
    activeOrder = orderID;
    if (activity != null) {
        activeActivity = activity;
        if (subActivity != null) {
            activeSubActivity = subActivity;
        }
    }
}

// Call MAM if data has never been persisted
if (mofs.read(Constants.MOFS_KEY_MAM_DATA) == null){
    // Set value in MultiObjectFileStorage that MAM is called from MTT
    mofs.write(Constants.MOFS_KEY_MAM_CALLED_FROM_MTS,Constants.VALUE_TRUE);
    return f.findForward(FORWARD_CALL_MAM);
} else{
    ((GridAdapter)naming.lookup(GridAdapter.class.getName())).resetBaseGrid();
}
return super.onLoad(f);
/**
 * @return
 */
public static String getActiveActivity() {
    return activeActivity;
}

/**
 * @return
 */
public static String getActiveOrder() {
    return activeOrder;
}

/**
 * @return
 */
public static String getClientURLText() {
    return clientURLText;
}
public static String getActiveSubActivity() {
    return activeSubActivity;
}

/**
 * @param string
 */
public static void setActiveActivity(String string) {
    activeActivity = string;
}

/**
 * @param string
 */
public static void setActiveOrder(String string) {
    activeOrder = string;
}

/**
 * @param string
 */
public static void setActiveSubActivity(String string) {
    activeSubActivity = string;
}

7.3.2 ZGridAdapterImpl

package zcom.sap.mbs.mts.model.grid.impl;

import java.util.Iterator;
import java.util.List;
import java.util.Map;
import zcom.sap.mbs.common.integration.ChildObject;
import zcom.sap.mbs.common.integration.Constants;
import zcom.sap.mbs.common.integration.ParentObject;
import zcom.sap.mbs.mts.home.control.impl.ZStartupController;

import com.sap.ip.me.api.services.MultiObjectFileStorage;
import com.sap.ip.me.api.user.UserManager;
import com.sap.mbs.mts.common.model.EmployeeModel;
public class ZGridAdapterImpl extends GridAdapterImpl {

    private List allFieldsBasedOnRepository;
    private Map allFieldsBasedOnFieldSelection;

    public ZGridAdapterImpl() {
    }

    public void initializeBaseGrid() throws DBException {
        super.initializeBaseGrid();
        try {
            EmployeeModel user = GridAdapterImpl.managerEmployee.getEmployee(MTTUserManager.getInstance().getLoggedOnUser());
            MultiObjectFileStorage mofs = MultiObjectFileStorage.createInstance(UserManager.getInstance().getCurrentUser().getUniqueID());
            List orders = (List) mofs.read(Constants.MOFS_KEY_MAM_DATA);

            // If MAM has persisted data and MTS has data
            if (orders != null && orders.size() > 0 && !user.getPERNR().equals(String.valueOf(!))){
                allFieldsBasedOnRepository = FieldSelectionManager.getAllFieldsBasedOnRepository(user);
                allFieldsBasedOnFieldSelection = FieldSelectionManager.getMergedFieldSelection(user);
                IRecord baseRecord = TEAdapter.newTypedRecord("recordversion");

                // Check if there is an order set to pre-populate time sheet with MAM
                Orders

                String activeOrderId = ZStartupController.getActiveOrder();
                String activeActivityId = ZStartupController.getActiveActivity();
                String activeSubActivityId = ZStartupController.getActiveSubActivity();

                if (activeOrderId != null) {
                    int currentRow = 0;
                    Iterator itOrder = orders.iterator();
                    // Fetch all the order
                }
        }
    }
}
while (itOrder.hasNext()) {
    ParentObject order = (ParentObject)itOrder.next();
    String orderId = order.getId();  
    // Keep only the one that corresponds to the activeOrderId
    if (orderId.equals(activeOrderId)) {
        // If no activity defined, create a template for all
        if (activeActivityId == null) {
            Iterator itAct = order.getChildren().iterator();
            while (itAct.hasNext()) {
                ChildObject activity = (ChildObject)itAct.next();
                itSubAct = activity.getChildren().iterator();
                // If Activity contains Sub
                if (itSubAct.hasNext()) {
                    while (itSubAct.hasNext()) {
                        ChildObject subActivity = (ChildObject)itSubAct.next();
                        createTemplate(orderId, activity.getId(), subActivity.getId(), user, (TypedRecord) baseRecord.getClone(), currentRow++);
                    }
                } else {
                    createTemplate(orderId, activity.getId(), null, user, (TypedRecord) baseRecord.getClone(), currentRow++);
                }
            }
        } else {
            if (activeSubActivityId == null) {
                Iterator itActivity = order.getChildren().iterator();
                while (itActivity.hasNext()) {
                    ChildObject activity = (ChildObject)itActivity.next();
                    TypedRecord recordVersion = (TypedRecord) baseRecord.getClone();
protected void createTemplate(String orderId, String activityId, String subActivityId, EmployeeModel user, TypedRecord recVersion, int index) throws Exception {
    recVersion.setField(ICatsField.RAUFNR, orderId);
    if (activityId != null) {
        recVersion.setField(ICatsField.VORNR, activityId);
    }
    if (subActivityId != null) {
        recVersion.setField(ICatsField.UVORN, subActivityId);
    }
    try {
        createTemplate(orderId, activityId, null, user, recVersion, currentRow++);
    } catch (Exception e) {
        TracePrintStream.println(e, TracePrintStream.TRACELEVEL_ERROR);
        throw new DBException(e);
    }
}
List fsValues = getFieldsAndValues(recVersion, allFieldsBasedOnRepository);

GridAdapterImpl.baseGrid.addTemplateElement(user,createFieldSelectionValues(fsValues,allFieldsBasedOnFieldSelection),index);
}
}

7.3.3 ZPicklistProcessImpl

```java
package zcom.sap.mbs.mts.picklist.process.impl;

import java.util.ArrayList;
import java.util.Collection;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import java.util.Map;
import com.sap.ip.me.api.services.MultiObjectFileStorage;
import com.sap.ip.me.api.user.UserManager;

import com.sap.mbs.mts.common.structure.ICatsField;
import com.sap.mbs.mtt.application.exception.DBException;
import com.sap.mbs.mtt.application.model.MTTUser;

import zcom.sap.mbs.common.integration.ChildObject;
import zcom.sap.mbs.common.integration.ParentObject;


public class ZPicklistProcessImpl extends PicklistProcessImpl {

    public static final String MOFS_KEY_MAM_DATA = "MAM_DATA";

    public ZPicklistProcessImpl() {
    }

    public Collection getValues(String fieldName, MTTUser forUser) throws DBException {
        System.out.println("in " + getClass().getName());
        MultiObjectFileStorage mofs = MultiObjectFileStorage.createInstance(UserManager.getInstance().getCurrentUser().getUniqueID());
        List allObjects = (List) mofs.read(MOFS_KEY_MAM_DATA);
```
Collection result;

if (ICatsField.RAUFNR.equals(fieldName) && allObjects != null && allObjects.size() > 0) {
    result = new ArrayList();

    Map picklistEntry = null;

    Iterator itOrders = allObjects.iterator();
    while (itOrders.hasNext()) {
        Object nadia = (Object)itOrders.next();
        System.out.println(nadia.getClass().getName());
        ParentObject order = (ParentObject)itOrders.next();
        String orderId = order.getId();
        String orderDesc = order.getField("description").getValue();
        Iterator itActivity = order.getChildrenByName("activity").iterator();
        picklistEntry = new HashMap();
        picklistEntry.put("RAUFNR", orderId);
        picklistEntry.put("RAUFNR_TXT", orderDesc);
        picklistEntry.put("VORNR", ")
        picklistEntry.put("VORNR_TXT", "");
        result.add(picklistEntry);

        while (itActivity.hasNext()) {
            ChildObject activity = (ChildObject)itActivity.next();
            String activityId = activity.getId();
            String activityDesc = activity.getField("description").getValue();
            picklistEntry = new HashMap();
            picklistEntry.put("RAUFNR", activityId);
            picklistEntry.put("RAUFNR_TXT", activityDesc);
            picklistEntry.put("VORNR", activityId);
            picklistEntry.put("VORNR_TXT", activityDesc);
            result.add(picklistEntry);

            Iterator itSubActivity = activity.getChildrenByName("subActivity").iterator();
            while (itSubActivity.hasNext()) {
                ChildObject subActivity = (ChildObject)itSubActivity.next();
                picklistEntry = new HashMap();
                picklistEntry.put("RAUFNR", orderId);
                picklistEntry.put("RAUFNR_TXT", orderDesc);
                picklistEntry.put("VORNR", activityId);
                picklistEntry.put("VORNR_TXT", activityDesc);
                result.add(picklistEntry);
            }
        }
    }
}
picklistEntry.put("VORNR_TXT", activityDesc);
picklistEntry.put("UVORN", subActivity.getId());
result.add(picklistEntry);
}
}
}
else {
    result = super.getValues(fieldName, forUser);
}
return result;
}

7.3.4  zstartup.jsp

<HTML>
<body onload="javascript:document.location.replace('/MAM/start');">
</body>
</HTML>

Where “MAM” is the name of the xMAM Mobile Application Component deployed locally.

7.3.5  zstartup.view

<?xml version="1.0" encoding="UTF-8"?>

<!-- NOTE: .view files have a 1 to 1 relationship with View Controllers (see related View Controller for details) -->

<view screen="/home/zstartup.jsp"
controller="zcom.sap.mbs.mts.home.control.impl.ZStartupController"
parent_view="startup">
  <event name="onLoad" inherit="true">
    <forward name="callMAM" view="zstartup"/>
  </event>
</view>

7.3.6  view.configure

home.startup=home.zstartup
7.3.7 zcore.configure

com.sap.mbs.mts.model.grid.GridAdapter=zcom.sap.mbs.mts.model.grid.impl.ZGridAdapterImpl