private void createStep(

MPAState mpaState, MPAOConstants.Step step, MPAOConstants.StepStatus stepStatus, MPAOConstants.StepReason stepReason)

throws ServiceException

{

addStep(mpaState, step, stepStatus, stepReason, MPAOConstants.ObjectType.Cust, mpaState.getCurrentApplicantId());

}

try

{

// Use AddReplace, since in most instances it is appropriate to replace an

// existing step.

MPAOMappingHelper.addReplaceStep(mpaState, step, stepStatus, stepReason, objType, objSeqNum);

}

catch (FactoryException e1)

{

throw new ServiceException(MPAOConstants.INSTANTIATION\_FAILURE, ErrorCode.INSTANTIATION\_FAILED, e1);

}

TmpBussProcessInfoType tbpi = mpaState.getState().getTmpBussProcessInfo();

if (tbpi == null)

{

tbpi = (TmpBussProcessInfoType) Factory.getInstanceByClass(TmpBussProcessInfoType.class);

}

TmpStatusInfoType tsi = tbpi.getTmpStatusInfo();

if (tsi == null)

{

tsi = (TmpStatusInfoType) Factory.getInstanceByClass(TmpStatusInfoType.class);

}

StepType newStep = buildStep(mpaState, step, stepStatus, stepReason, objType, objSeqNum);

boolean stepFound = false;

for (int i = 0; i < tsi.getStep().size(); i++)

{

StepType tempStep = (StepType) tsi.getStep().get(i);

if (tempStep.getStepTypeElem().equals(newStep.getStepTypeElem())

&& tempStep.getObjType().equals(newStep.getObjType())

&& tempStep.getObjSeqNum() == newStep.getObjSeqNum())

{

// replace the old step (but keep the old seqNum)

newStep.setSeqNum(tempStep.getSeqNum());

tsi.getStep().set(i, newStep);

stepFound = true;

break;

}

}

if (!stepFound)

{

tsi.getStep().add(newStep);

}

tbpi.setTmpStatusInfo(tsi);

mpaState.getState().setTmpBussProcessInfo(tbpi);

StepType currentApplicantStep = MPAOMappingHelper.getStep(

mpaState,

Step.CustProfileProcess,

MPAOConstants.ObjectType.Cust,

applicant.getAddPartyInfo().getSeqNum());

public static StepType getStep(

final MPAState mpaState, final MPAOConstants.Step step, final MPAOConstants.ObjectType objType, final long objSeqNum)

{

return getStep(mpaState.getState(), step, objType, objSeqNum);

}

final TmpBussProcessInfoType tbpi = sbpi.getTmpBussProcessInfo();

if (tbpi != null)

{

if (tbpi.getTmpStatusInfo() != null)

{

final List steps = tbpi.getTmpStatusInfo().getStep();

if (step != null)

{

final Iterator iter = steps.iterator();

while (iter.hasNext())

{

final StepType thisStep = (StepType) iter.next();

if (thisStep.getObjType() != null

&& thisStep.getObjType().equals(objType.name())

&& thisStep.getObjSeqNum() == objSeqNum

&& thisStep.getStepTypeElem() != null

&& thisStep.getStepTypeElem().equals(step.name()))

{

return thisStep;

}

}

}

}

}

return null;

import com.satya.MPAOMapping;

import com.satya.PreparedDataForMPAO;

public class Test {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

MPAOMapping mapping = new MPAOMapping();

PreparedDataForMPAO mpao = new PreparedDataForMPAO();

mapping.getData(mpao);

}

}

package com.satya;

public class MPAO {

int roll;

String name;

String subj;

public int getRoll() {

return roll;

}

public void setRoll(int roll) {

this.roll = roll;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getSubj() {

return subj;

}

public void setSubj(String subj) {

this.subj = subj;

}

}

package com.satya;

import java.util.ArrayList;

public class MPAOMapping {

ArrayList<MPAO> list = new ArrayList<MPAO>();

public void getData(PreparedDataForMPAO mpao)

{

mpao.setApp(createData());

MPAO[] app = mpao.getApp();

int len = app.length;

for(int i=0;i<len;i++)

{

System.out.println(app[i].name);

System.out.println(app[i].roll);

System.out.println(app[i].subj);

}

}

private MPAO[] createData() {

// TODO Auto-generated method stub

MPAO mpao = new MPAO();

MPAO mpao1 = new MPAO();

mpao.setRoll(1);

mpao.setName("vicky");

mpao.setSubj("Java");

mpao1.setRoll(1);

mpao1.setName("vicky");

mpao1.setSubj("Java");

list.add(mpao);

list.add(mpao1);

return list.toArray(new MPAO[list.size()]);

}

}

package com.satya;

public class PreparedDataForMPAO {

private MPAO[] app;

public void setApp(MPAO[] app) {

this.app = app;

}

public MPAO[] getApp() {

// TODO Auto-generated method stub

return app;

}

}