/\*

\* @(#) OrderServiceEx.java May 5, 2018 4:46:50 PM

\*

\* Copyright 2018 Rockwell Automation, Inc. All rights reserved.

\* Rockwell Automation PROPRIETARY/CONFIDENTIAL. Use is subject to license terms.

\*/package com.rockwell.autosuite.mes.services.om.impl;

import java.math.BigDecimal;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Iterator;

import java.util.List;

import java.util.Map;

import java.util.Set;

import java.util.Vector;

import com.datasweep.compatibility.client.Area;

import com.datasweep.compatibility.client.DatasweepException;

import com.datasweep.compatibility.client.Error;

import com.datasweep.compatibility.client.Order;

import com.datasweep.compatibility.client.OrderItem;

import com.datasweep.compatibility.client.Part;

import com.datasweep.compatibility.client.Response;

import com.datasweep.compatibility.client.Station;

import com.datasweep.compatibility.client.Unit;

import com.datasweep.compatibility.client.UserSequence;

import com.datasweep.compatibility.client.UserSequenceValue;

import com.datasweep.compatibility.client.WorkCenter;

import com.datasweep.compatibility.ui.Time;

import com.datasweep.plantops.common.constants.filtering.IATRowFilterAttributes;

import com.datasweep.plantops.common.constants.filtering.IFilterSortOrders;

import com.rockwell.autosuite.mes.constants.common.IModuleName;

import com.rockwell.autosuite.mes.constants.common.OrderItemUDAName;

import com.rockwell.autosuite.mes.constants.common.OrderUDAName;

import com.rockwell.autosuite.mes.constants.common.UnitUDAName;

import com.rockwell.autosuite.mes.constants.om.BDCAreaName;

import com.rockwell.autosuite.mes.constants.om.OrderSourceStatus;

import com.rockwell.autosuite.mes.constants.om.ReworkOperType;

import com.rockwell.autosuite.mes.constants.om.SubLineType;

import com.rockwell.autosuite.mes.constants.om.TakeInOutStatus;

import com.rockwell.autosuite.mes.model.om.batchsuborder.IMESASOMBatchSubOrder;

import com.rockwell.autosuite.mes.model.om.batchsuborder.MESASOMBatchSubOrder;

import com.rockwell.autosuite.mes.model.om.bdcattrgroup.IMESASOMBDCAttrGroup;

import com.rockwell.autosuite.mes.model.om.bdcattrgroup.MESASOMBDCAttrGroupItem;

import com.rockwell.autosuite.mes.model.om.bdccontinuous.IMESASOMBDCContinuous;

import com.rockwell.autosuite.mes.model.om.bdccontinuous.IMESASOMBDCContinuousFilter;

import com.rockwell.autosuite.mes.model.om.bdccontinuous.MESASOMBDCContinuousFilter;

import com.rockwell.autosuite.mes.model.om.bdcholdvehicle.IMESASOMBDCHoldVehicle;

import com.rockwell.autosuite.mes.model.om.bdcholdvehicle.IMESASOMBDCHoldVehicleFilter;

import com.rockwell.autosuite.mes.model.om.bdcholdvehicle.MESASOMBDCHoldVehicleFilter;

import com.rockwell.autosuite.mes.model.om.bdcinoutlog.IMESASOMBDCInOutLog;

import com.rockwell.autosuite.mes.model.om.bdcinoutlog.MESASOMBDCInOutLog;

import com.rockwell.autosuite.mes.model.om.bdcoutqueue.IMESASOMBDCOutQueue;

import com.rockwell.autosuite.mes.model.om.bdcoutqueue.IMESASOMBDCOutQueueFilter;

import com.rockwell.autosuite.mes.model.om.bdcoutqueue.MESASOMBDCOutQueue;

import com.rockwell.autosuite.mes.model.om.bdcoutqueue.MESASOMBDCOutQueueFilter;

import com.rockwell.autosuite.mes.model.om.bdcoutqueue.MESGeneratedASOMBDCOutQueue;

import com.rockwell.autosuite.mes.model.om.bdcratio.IMESASOMBDCRatio;

import com.rockwell.autosuite.mes.model.om.bdcratio.IMESASOMBDCRatioFilter;

import com.rockwell.autosuite.mes.model.om.bdcratio.MESASOMBDCRatioFilter;

import com.rockwell.autosuite.mes.model.om.bdcspace.IMESASOMBDCSpace;

import com.rockwell.autosuite.mes.model.om.bdcspace.IMESASOMBDCSpaceFilter;

import com.rockwell.autosuite.mes.model.om.bdcspace.MESASOMBDCSpaceFilter;

import com.rockwell.autosuite.mes.model.om.bdcstock.IMESASOMBDCStock;

import com.rockwell.autosuite.mes.model.om.bdcstock.IMESASOMBDCStockFilter;

import com.rockwell.autosuite.mes.model.om.bdcstock.MESASOMBDCStockFilter;

import com.rockwell.autosuite.mes.model.om.bdcstock.MESGeneratedASOMBDCStock;

import com.rockwell.autosuite.mes.model.om.biwsuborder.IMESASOMBIWSubOrder;

import com.rockwell.autosuite.mes.model.om.biwsuborder.MESASOMBIWSubOrder;

import com.rockwell.autosuite.mes.model.om.biwsuborder.MESASOMBIWSubOrderFilter;

import com.rockwell.autosuite.mes.model.om.electroorder.IMESASOMElectroOrder;

import com.rockwell.autosuite.mes.model.om.electroorder.IMESASOMElectroOrderFilter;

import com.rockwell.autosuite.mes.model.om.electroorder.MESASOMElectroOrderFilter;

import com.rockwell.autosuite.mes.model.om.featuregroup.IMESASOMFeatureGroup;

import com.rockwell.autosuite.mes.model.om.jobnosubline.IMESASOMJobNoSubLine;

import com.rockwell.autosuite.mes.model.om.jobnosubline.MESASOMJobNoSubLineFilter;

import com.rockwell.autosuite.mes.model.om.orderbom.IMESASOMOrderBom;

import com.rockwell.autosuite.mes.model.om.orderbom.IMESASOMOrderBomFilter;

import com.rockwell.autosuite.mes.model.om.orderbom.MESASOMOrderBomFilter;

import com.rockwell.autosuite.mes.model.om.orderbp.IMESASOMOrderBP;

import com.rockwell.autosuite.mes.model.om.orderbp.MESASOMOrderBP;

import com.rockwell.autosuite.mes.model.om.orderproperty.IMESASOMOrderProperty;

import com.rockwell.autosuite.mes.model.om.orderproperty.MESASOMOrderPropertyFilter;

import com.rockwell.autosuite.mes.model.om.orderrouteplan.IMESASOMOrderRoutePlanFilter;

import com.rockwell.autosuite.mes.model.om.orderrouteplan.MESASOMOrderRoutePlanFilter;

import com.rockwell.autosuite.mes.model.om.partbp.IMESASOMPartBP;

import com.rockwell.autosuite.mes.model.om.partbp.IMESASOMPartBPFilter;

import com.rockwell.autosuite.mes.model.om.partbp.MESASOMPartBP;

import com.rockwell.autosuite.mes.model.om.partbp.MESASOMPartBPFilter;

import com.rockwell.autosuite.mes.model.om.production.IMESASOMPlanProduction;

import com.rockwell.autosuite.mes.model.om.production.MESASOMPlanProductionFilter;

import com.rockwell.autosuite.mes.model.om.rfidandepcbindinglog.IMESASOMRFIDAndEPCBinding;

import com.rockwell.autosuite.mes.model.om.rfidandepcbindinglog.IMESASOMRFIDAndEPCBindingFilter;

import com.rockwell.autosuite.mes.model.om.rfidandepcbindinglog.MESASOMRFIDAndEPCBindingFilter;

import com.rockwell.autosuite.mes.model.om.rwareavehicle.IMESASOMRWAreaVehicle;

import com.rockwell.autosuite.mes.model.om.rwareavehicle.IMESASOMRWAreaVehicleFilter;

import com.rockwell.autosuite.mes.model.om.rwareavehicle.MESASOMRWAreaVehicle;

import com.rockwell.autosuite.mes.model.om.rwareavehicle.MESASOMRWAreaVehicleFilter;

import com.rockwell.autosuite.mes.model.om.rwareavehiclelog.IMESASOMRWAreaVehicleLog;

import com.rockwell.autosuite.mes.model.om.rwareavehiclelog.MESASOMRWAreaVehicleLog;

import com.rockwell.autosuite.mes.model.om.skidcodebindinglog.IMESASOMSkidCodeBindingLog;

import com.rockwell.autosuite.mes.model.om.skidcodebindinglog.IMESASOMSkidCodeBindingLogFilter;

import com.rockwell.autosuite.mes.model.om.skidcodebindinglog.MESASOMSkidCodeBindingLog;

import com.rockwell.autosuite.mes.model.om.skidcodebindinglog.MESASOMSkidCodeBindingLogFilter;

import com.rockwell.autosuite.mes.model.om.sqd.IMESASOMSQD;

import com.rockwell.autosuite.mes.model.om.sqd.MESASOMSQDFilter;

import com.rockwell.autosuite.mes.model.om.steelcodebinding.IMESASOMSteelCodeBinding;

import com.rockwell.autosuite.mes.model.om.steelcodebinding.IMESASOMSteelCodeBindingFilter;

import com.rockwell.autosuite.mes.model.om.steelcodebinding.MESASOMSteelCodeBinding;

import com.rockwell.autosuite.mes.model.om.steelcodebinding.MESASOMSteelCodeBindingFilter;

import com.rockwell.autosuite.mes.model.om.steelcodebindinglog.IMESASOMSteelCodeBindingLog;

import com.rockwell.autosuite.mes.model.om.steelcodebindinglog.MESASOMSteelCodeBindingLog;

import com.rockwell.autosuite.mes.model.om.sublinetype.IMESASSMSubLineType;

import com.rockwell.autosuite.mes.model.om.sublinetype.IMESASSMSubLineTypeFilter;

import com.rockwell.autosuite.mes.model.om.sublinetype.MESASSMSubLineType;

import com.rockwell.autosuite.mes.model.om.sublinetype.MESASSMSubLineTypeFilter;

import com.rockwell.autosuite.mes.model.om.syncsuborder.MESASOMSyncSubOrder;

import com.rockwell.autosuite.mes.model.om.synctakeinoutstatus.IMESASOMSyncTakeInOutStatus;

import com.rockwell.autosuite.mes.model.om.synctakeinoutstatus.IMESASOMSyncTakeInOutStatusFilter;

import com.rockwell.autosuite.mes.model.om.synctakeinoutstatus.MESASOMSyncTakeInOutStatus;

import com.rockwell.autosuite.mes.model.om.synctakeinoutstatus.MESASOMSyncTakeInOutStatusFilter;

import com.rockwell.autosuite.mes.model.om.takeinout.IMESASOMTakeInOut;

import com.rockwell.autosuite.mes.model.om.takeinout.IMESASOMTakeInOutFilter;

import com.rockwell.autosuite.mes.model.om.takeinout.MESASOMTakeInOut;

import com.rockwell.autosuite.mes.model.om.takeinout.MESASOMTakeInOutFilter;

import com.rockwell.autosuite.mes.model.om.takeinoutbcodebind.IMESASOMTakeInOutBCodeBind;

import com.rockwell.autosuite.mes.model.om.takeinoutbcodebind.IMESASOMTakeInOutBCodeBindFilter;

import com.rockwell.autosuite.mes.model.om.takeinoutbcodebind.MESASOMTakeInOutBCodeBind;

import com.rockwell.autosuite.mes.model.om.takeinoutbcodebind.MESASOMTakeInOutBCodeBindFilter;

import com.rockwell.autosuite.mes.model.om.takeinouthistory.MESASOMTakeOutHistory;

import com.rockwell.autosuite.mes.model.om.takeinoutstatus.IMESASOMTakeInOutStatus;

import com.rockwell.autosuite.mes.model.om.takeinoutstatus.IMESASOMTakeInOutStatusFilter;

import com.rockwell.autosuite.mes.model.om.takeinoutstatus.MESASOMTakeInOutStatus;

import com.rockwell.autosuite.mes.model.om.takeinoutstatus.MESASOMTakeInOutStatusFilter;

import com.rockwell.autosuite.mes.model.om.yearcodemapping.IMESASOMYearCodeMapping;

import com.rockwell.autosuite.mes.model.om.yearcodemapping.IMESASOMYearCodeMappingFilter;

import com.rockwell.autosuite.mes.model.om.yearcodemapping.MESASOMYearCodeMappingFilter;

import com.rockwell.autosuite.mes.model.sm.calendar.IMESASSMCalendar;

import com.rockwell.autosuite.mes.model.sm.calendar.MESASSMCalendar;

import com.rockwell.autosuite.mes.model.sm.calendar.MESASSMCalendarFilter;

import com.rockwell.autosuite.mes.model.sm.platform.IMESASSMPlatform;

import com.rockwell.autosuite.mes.model.sm.stationtype.IMESASSMStationTypeFilter;

import com.rockwell.autosuite.mes.model.sm.stationtype.MESASSMStationTypeFilter;

import com.rockwell.autosuite.mes.model.subpartbinding.IMESASOMSubPartBinding;

import com.rockwell.autosuite.mes.model.subpartbinding.IMESASOMSubPartBindingFilter;

import com.rockwell.autosuite.mes.model.subpartbinding.MESASOMSubPartBinding;

import com.rockwell.autosuite.mes.model.subpartbinding.MESASOMSubPartBindingFilter;

import com.rockwell.autosuite.mes.services.common.ifc.IMessage;

import com.rockwell.autosuite.mes.services.om.ifc.IOrderServiceEx;

import com.rockwell.autosuite.mes.services.sm.ifc.ISystemService;

import com.rockwell.autosuite.mes.services.sm.ifc.ISystemServiceEx;

import com.rockwell.autosuite.mes.utilities.common.ShopMasterManager;

import com.rockwell.autosuite.mes.utilities.common.TimeHelper;

import com.rockwell.autosuite.mes.utilities.om.OrderHelper;

import com.rockwell.autosuite.mes.utilities.om.OrderHelperEx;

import com.rockwell.autosuite.mes.utilities.om.OrderZone;

import com.rockwell.autosuite.mes.utilities.sm.SystemHelperEx;

import com.rockwell.common.utilities.LogUtility;

import com.rockwell.mes.commons.base.ifc.configuration.MESConfiguration;

import com.rockwell.mes.commons.base.ifc.exceptions.MESException;

import com.rockwell.mes.commons.base.ifc.services.PCContext;

import com.rockwell.mes.commons.base.ifc.services.ServiceFactory;

import com.rockwell.mes.commons.base.ifc.utility.StringUtilsEx;

public class OrderServiceEx extends OrderService implements IOrderServiceEx

{

private static String MODEL\_NAME = IModuleName.MODULE\_OM;

private static final String BDC\_OUT\_QUEUE\_CALC\_NUMBER\_PROPERTY = "BDC\_OUT\_QUEUE\_CALC\_NUMBER";

private static final Long BDC\_OUT\_QUEUE\_CALC\_NUMBER = 20l;

private static final String BDC\_OUT\_QUEUE\_AUTO\_CALC\_PROPERTY = "BDC\_OUT\_QUEUE\_AUTO\_CALC";

private ISystemService getSystemServiceEx()

{

return ServiceFactory.getService("SystemServiceEx", ISystemService.class);

}

@Override

protected String getYearCode(Order workOrder) throws Exception

{

String currentCalendarYearStr = (String) workOrder.getUDA("model\_year");

if(null == currentCalendarYearStr || "".equals(currentCalendarYearStr))

{

throw new Exception("Get model year from uda\_order is null. Order : " + workOrder.getOrderNumber());

}

String yearCode = null;

IMESASOMYearCodeMappingFilter yearCodeMappingFilter = new MESASOMYearCodeMappingFilter();

yearCodeMappingFilter.forCalendaryearEqualTo(currentCalendarYearStr);

List<IMESASOMYearCodeMapping> yearCodeList = yearCodeMappingFilter.getFilteredObjects();

if (yearCodeList != null && yearCodeList.size() > 0)

{

IMESASOMYearCodeMapping yearCodeMapping = yearCodeList.get(0);

yearCode = yearCodeMapping.getYearcode();

}

else

{

throw new Exception("Can't get year code by year " + currentCalendarYearStr +" . Order : " + workOrder.getOrderNumber());

}

return yearCode;

}

public Unit getTCFShopUnit(Order order)

{

List<Area> assemblyShops = ShopMasterManager.getInstance().getTCFShops();

Vector allUnits = order.getAllUnits();

for (Object object : allUnits)

{

Unit unit = (Unit) object;

try

{

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

{

String shop = assemblyShops.get(i).getName();

if (shop != null && shop.equals(((Area) unit.getUDA(UnitUDAName.SHOP)).getName()))

{

return unit;

}

}

}

catch (DatasweepException e)

{

throw new RuntimeException(e);

}

}

return null;

}

@Override

public IMESASOMBDCStock getBDCStockObj(Order order,String area) throws DatasweepException

{

IMESASOMBDCStock bdcStock = null;

IMESASOMBDCStockFilter bdcStockFilter = new MESASOMBDCStockFilter();

bdcStockFilter.forOrderEqualTo(order);

bdcStockFilter.forAreaEqualTo(area);

List<IMESASOMBDCStock> bdcStockList = bdcStockFilter.getFilteredObjects();

if(bdcStockList.size()>0)

{

bdcStock = bdcStockList.get(0);

}

return bdcStock;

}

@Override

public Response getBDCStockObjPnuts(Order order,String area)

{

Response resp = new Response();

try

{

IMESASOMBDCStock bdcStock = getBDCStockObj(order,area);

resp.setResult(bdcStock);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public IMESASOMTakeInOut getTakeInOutObj(String bsn,String orderType) throws DatasweepException

{

IMESASOMTakeInOut takeInOut = null;

IMESASOMTakeInOutFilter takeInOutFilter = new MESASOMTakeInOutFilter();

takeInOutFilter.forBsnEqualTo(bsn);

takeInOutFilter.forOrdertypeEqualTo(orderType);

List<IMESASOMTakeInOut> takeInOutList = takeInOutFilter.getFilteredObjects();

if(takeInOutList.size()>0)

{

takeInOut = takeInOutList.get(0);

}

return takeInOut;

}

@Override

public Response getTakeInOutObjPnuts(String bsn,String orderType)

{

Response resp = new Response();

try

{

IMESASOMTakeInOut takeInOut = getTakeInOutObj(bsn, orderType);

resp.setResult(takeInOut);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void takeIn(Order order,String workCenterStr,String orderType,Time operTime) throws DatasweepException, MESException

{

String functionName = "takeIn(Order order,String workCenterStr,String orderType,Time operTime)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

if (StringUtilsEx.isBlank(workCenterStr))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN workCenterStr is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "workCenterStr" });

}

if (StringUtilsEx.isBlank(orderType))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN workCenterStr is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderType" });

}

if (operTime == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN operTime is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "operTime" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],workCenterStr:["+ workCenterStr + "],orderType:["+ orderType + "];";

String bsn = OrderHelper.getBsn(order);

IMESASOMTakeInOut takeInOut = getTakeInOutObj(bsn, orderType);

if(takeInOut == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN operTime is null!KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("The vehicle did not take out.bsn["+bsn+"]");

}

else

{

Long stauts = takeInOut.getStatus();

if(stauts==1l)

{

throw new MESException("The vehicle has been take in.bsn["+bsn+"]");

}

}

takeInOut.setStatus(1L);

WorkCenter workCenter = PCContext.getFunctions().getWorkCenterByName(workCenterStr);

takeInOut.setOperationstation(workCenter);

takeInOut.setOperationtime(operTime);

takeInOut.setOperationuser(PCContext.getFunctions().getCurrentUser().getName());

Response response = takeInOut.save(null, null, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMTakeInOut success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMTakeInOut fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMTakeInOut", keyWord });

}

saveTakeOutHistory(order,workCenterStr,orderType,operTime,TakeInOutStatus.IN);

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response takeInPnuts(Order order,String workCenterStr,String orderType,Time operTime)

{

Response resp = new Response();

try

{

takeIn(order, workCenterStr, orderType, operTime);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void takeOut(Order order,String workCenterStr,String orderType,Time operTime) throws DatasweepException, MESException

{

String functionName = "takeOut(Order order,String workCenterStr,String orderType,Time operTime)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

if (StringUtilsEx.isBlank(workCenterStr))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN workCenterStr is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "workCenterStr" });

}

if (StringUtilsEx.isBlank(orderType))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN workCenterStr is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderType" });

}

if (operTime == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN operTime is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "operTime" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],workCenterStr:["+ workCenterStr + "],orderType:["+ orderType + "];";

String bsn = OrderHelper.getBsn(order);

IMESASOMTakeInOut takeInOut = getTakeInOutObj(bsn, orderType);

if(takeInOut != null)

{

Long stauts = takeInOut.getStatus();

if(stauts==TakeInOutStatus.OUT)

{

LogUtility.logError(

MODEL\_NAME, "The vehicle has been take out, KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("The vehicle has been take out, KeyWord<" + keyWord + ">");

}

}

else

{

takeInOut = new MESASOMTakeInOut();

takeInOut.setBsn(bsn);

takeInOut.setOrdertype(orderType);

}

takeInOut.setStatus(TakeInOutStatus.OUT);

WorkCenter workCenter = PCContext.getFunctions().getWorkCenterByName(workCenterStr);

takeInOut.setOperationstation(workCenter);

takeInOut.setOperationtime(operTime);

takeInOut.setOperationuser(PCContext.getFunctions().getCurrentUser().getName());

Response response = takeInOut.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMTakeInOut success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMTakeInOut fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMTakeInOut", keyWord });

}

saveTakeOutHistory(order,workCenterStr,orderType,operTime,TakeInOutStatus.OUT);

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response takeOutPnuts(Order order,String workCenterStr,String orderType,Time operTime)

{

Response resp = new Response();

try

{

takeOut(order, workCenterStr, orderType, operTime);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void saveTakeOutHistory(Order order,String workCenterStr,String orderType,Time operTime,Long status) throws DatasweepException, MESException

{

String functionName = "saveTakeOutHistory(Order order,String workCenterStr,String orderType,Time operTime,Long status)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

if (StringUtilsEx.isBlank(workCenterStr))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN workCenterStr is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "workCenterStr" });

}

if (StringUtilsEx.isBlank(orderType))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN orderType is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderType" });

}

if (operTime == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN operTime is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "operTime" });

}

if (status == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN status is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "status" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],workCenterStr:["+ workCenterStr + "],orderType:["+ orderType + "];";

MESASOMTakeOutHistory takeOutHistory = new MESASOMTakeOutHistory();

String bsn = OrderHelper.getBsn(order);

String partNumber = (String) order.getUDA(OrderUDAName.PART\_NUMBER);

String platForm = (String) order.getUDA(OrderUDAName.MODEL);

takeOutHistory.setBsn(bsn);

takeOutHistory.setOrdertype(orderType);

takeOutHistory.setPlatform(platForm);

takeOutHistory.setPartnumber(partNumber);

takeOutHistory.setStatus(status);

takeOutHistory.setOperator(PCContext.getFunctions().getCurrentUser().getName());

takeOutHistory.setOpertime(operTime);

takeOutHistory.setOperstation(workCenterStr);

Response response= takeOutHistory.save(null,keyWord,null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMTakeOutHistory success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMTakeOutHistory fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMTakeOutHistory", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response saveTakeOutPnuts(Order order,String workCenterStr,String orderType,Time operTime,Long status)

{

Response resp = new Response();

try

{

saveTakeOutHistory(order, workCenterStr, orderType, operTime, status);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void saveBDCInOutLog(Order order, String area, String opType,Time opTime,String orderNumber,String bsn,String vin,String paintColor,Boolean isElectroOrder,String inType) throws DatasweepException, MESException

{

String functionName = "saveBDCInOutLog(Order order, String area, String opType)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (StringUtilsEx.isBlank(area))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN area is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "area" });

}

if (StringUtilsEx.isBlank(opType))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN opType is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "opType" });

}

if (StringUtilsEx.isBlank(orderNumber))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN orderNumber is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderNumber" });

}

if (StringUtilsEx.isBlank(vin))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN vin is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderNumber" });

}

String keyWord = "area:["+ area + "],OrderNumber:[" + orderNumber + "],vin:[" + vin + "],opType:["+ opType + "];";

IMESASOMBDCInOutLog bdcInOutLog = new MESASOMBDCInOutLog();

bdcInOutLog.setOrder(order);

bdcInOutLog.setArea(area);

bdcInOutLog.setOpertype(opType);

bdcInOutLog.setOpertime(opTime);

bdcInOutLog.setOrdernumber(orderNumber);

bdcInOutLog.setBsn(bsn);

bdcInOutLog.setVin(vin);

bdcInOutLog.setPaintcolor(paintColor);

bdcInOutLog.setIselectroorder(isElectroOrder);

bdcInOutLog.setIntype(inType);

Response response = bdcInOutLog.save(null, null, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMBDCInOutLog success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save IMESASOMBDCInOutLog fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "IMESASOMBDCInOutLog", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response saveBDCInOutLogPnuts(Order order, String area, String opType,Time opTime,String orderNumber,String bsn,String vin,String paintColor,Boolean isElectroOrder,String inType)

{

Response resp = new Response();

try

{

saveBDCInOutLog(order, area, opType, opTime, orderNumber, bsn, vin, paintColor, isElectroOrder,inType);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void deductingBreakPoint() throws MESException

{

String functionName = "deductingBreakPoint()";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

StringBuffer bufSql = new StringBuffer();

bufSql.append("select bp.atr\_key,bp.change\_no\_s,nvl(sum(qty\_d),0) from at\_as\_om\_partbp bp,");

bufSql.append(" at\_as\_om\_orderproperty dop,");

bufSql.append(" at\_as\_om\_orderproperty sop,");

bufSql.append(" at\_as\_om\_orderproperty op,");

bufSql.append(" at\_as\_om\_order\_bom ob ");

bufSql.append(" where bp.is\_deducting\_y=0");

bufSql.append(" and bp.start\_order\_54 = sop.order\_54");

bufSql.append(" and bp.deducting\_order\_54 = dop.order\_54");

bufSql.append(" and op.mix\_s>=sop.mix\_s");

bufSql.append(" and op.mix\_s<=dop.mix\_s");

bufSql.append(" and op.order\_54 = ob.work\_order\_i");

bufSql.append(" and bp.old\_part\_21=ob.part\_21");

bufSql.append(" group by bp.atr\_key,bp.change\_no\_s");

@SuppressWarnings("unchecked")

List<String[]> partBpList = PCContext.getFunctions().getArrayDataFromActive(

bufSql.toString());

LogUtility.logInfo(MODEL\_NAME, "deductingBreakPoint SQL:[ \n" + bufSql.toString() + "]",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

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

{

String[] rowData = partBpList.get(i);

LogUtility.logInfo(MODEL\_NAME, (i + 1) + ",key:[" + rowData[0] + ",change\_no:[" + rowData[1] + "],qty:[" + rowData[2] + "].",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

long partBpKey = Long.valueOf(rowData[0]);

MESASOMPartBP partBp = new MESASOMPartBP(partBpKey);

partBp.setIsdeducting(true);

BigDecimal surplusQty = BigDecimal.valueOf(Long.valueOf(rowData[2]));

surplusQty = surplusQty.subtract(partBp.getSurplusqty());

partBp.setSurplusqty(surplusQty);

Response response = partBp.save(null, rowData[1], null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMPartBP success,change\_no:[" + rowData[1] + "]",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMPartBP fail,change\_no:[" + rowData[1] + "],ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMPartBP", rowData[1] });

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response deductingBreakPointPnuts()

{

Response resp = new Response();

try

{

deductingBreakPoint();

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void calcBreakPoint(Order order) throws DatasweepException, MESException

{

String functionName = "calcBreakPoint(Order order)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "]";

IMESASOMOrderBomFilter orderBomFilter = new MESASOMOrderBomFilter();

orderBomFilter.forWorkorderEqualTo(order.getKey());

List<IMESASOMOrderBom> orderBomList = orderBomFilter.getFilteredObjects();

IMESASOMPartBPFilter partBpFilter = new MESASOMPartBPFilter();

partBpFilter.forDeductingorderNotEqualTo(order);

List<IMESASOMPartBP> partBpList = partBpFilter.getFilteredObjects();

for(IMESASOMPartBP partBp:partBpList)

{

String keyWordBp = keyWord + ",Changeno:["+ partBp.getChangeno() +"]";

for(IMESASOMOrderBom orderBom:orderBomList)

{

if(partBp.getOldpart().getKey() == orderBom.getPart().getKey())

{

String keyWordPart = keyWordBp + ",oldPartNumber[" + partBp.getOldpart().getPartNumber() + "]";

if(partBp.getSurplusqty().compareTo(BigDecimal.ZERO)==1)

{

LogUtility.logInfo(

MODEL\_NAME,"surplus qty greater than ZERO, keyWordPart<" + keyWordPart + ">" ,

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(partBp.getSurplusqty().compareTo(orderBom.getQty())<1)

{

partBp.setForecastorder(order);

LogUtility.logInfo(

MODEL\_NAME,"surplus qty less than bom qty, keyWordPart<" + keyWordPart + ">" ,

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

partBp.setSurplusqty(partBp.getSurplusqty().subtract(orderBom.getQty()));

}

else

{

LogUtility.logInfo(

MODEL\_NAME,"surplus qty less than ZERO, keyWordPart<" + keyWordPart + ">" ,

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

recordOrderBreakPoint(order, partBp, partBp.getOldpart(), partBp.getNewpart());

changeOrderBom(orderBom, partBp.getNewpart());

}

}

else if(partBp.getNewpart().getKey() == orderBom.getPart().getKey())

{

String keyWordPart = keyWordBp + ",newPartNumber[" + partBp.getNewpart().getPartNumber() + "]";

if(partBp.getSurplusqty().compareTo(BigDecimal.ZERO)==1)

{

LogUtility.logInfo(

MODEL\_NAME,"surplus qty greater than ZERO, keyWordPart<" + keyWordPart + ">" ,

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(partBp.getSurplusqty().compareTo(orderBom.getQty())<1)

{

partBp.setForecastorder(order);

LogUtility.logInfo(

MODEL\_NAME,"surplus qty less than bom qty, keyWordPart<" + keyWordPart + ">" ,

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

partBp.setSurplusqty(partBp.getSurplusqty().subtract(orderBom.getQty()));

recordOrderBreakPoint(order, partBp, partBp.getNewpart(), partBp.getOldpart());

changeOrderBom(orderBom, partBp.getOldpart());

}

else

{

LogUtility.logInfo(

MODEL\_NAME,"surplus qty less than ZERO, keyWordPart<" + keyWordPart + ">" ,

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

}

}

partBp.setDeductingorder(order);

Response response = partBp.save(null, keyWordBp, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMPartBP success,keyWordBp<" + keyWordBp + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMPartBP fail, keyWordBp<" + keyWordBp + ">"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMPartBP", keyWordBp });

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response calcBreakPointPnuts(Order order)

{

Response resp = new Response();

try

{

calcBreakPoint(order);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

private void recordOrderBreakPoint(Order order,IMESASOMPartBP partBp,Part oldPart,Part newPart) throws MESException

{

String functionName = "recordOrderBreakPoint(Order order,IMESASOMPartBP partBp,Part oldPart,Part newPart)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

if (partBp == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN partBp is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "partBp" });

}

if (oldPart == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN oldPart is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "oldPart" });

}

if (newPart == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "newPart" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],changeNo["+partBp.getChangeno()+"],oldPartNumber["+oldPart.getPartNumber()+"],newPartNumber["+newPart.getPartNumber()+"]";

IMESASOMOrderBP orderBp = new MESASOMOrderBP();

orderBp.setOrder(order);

orderBp.setPartbp(partBp);

orderBp.setOldpart(oldPart);

orderBp.setNewpart(newPart);

orderBp.setChangtime(PCContext.getFunctions().getDBTime());

Response response = orderBp.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMOrderBP success,keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMOrderBP fail, keyWord<" + keyWord + ">"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMOrderBP", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

private void changeOrderBom(IMESASOMOrderBom orderBom, Part newPart) throws MESException

{

String functionName = "recordOrderBreakPoint(Order order,IMESASOMPartBP partBp,Part oldPart,Part newPart)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (orderBom == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN orderBom is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderBom" });

}

if (newPart == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "newPart" });

}

String keyWord = "orderBomKey:[" + orderBom.getKey() + "],newPartNumber["+newPart.getPartNumber()+"]";

orderBom.setPart(newPart);

Response response = orderBom.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMOrderBom success,keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMOrderBom fail, keyWord<" + keyWord + ">"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMOrderBom", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Boolean isInReworkArea(Order order) throws DatasweepException

{

Boolean isIn = true;

if(getRWAreaVehicle(order) == null)

{

isIn = false;

}

return isIn;

}

@Override

public Response isInReworkAreaPnuts(Order order)

{

Response resp = new Response();

try

{

Boolean result = isInReworkArea(order);

resp.setResult(result);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public IMESASOMRWAreaVehicle getRWAreaVehicle(Order order) throws DatasweepException

{

IMESASOMRWAreaVehicle rwAreaVehicle = null;

IMESASOMRWAreaVehicleFilter filter = new MESASOMRWAreaVehicleFilter();

filter.forOrderEqualTo(order);

filter.setMaxRows(1);

List<IMESASOMRWAreaVehicle> rwAreaVehicleList = filter.getFilteredObjects();

if(rwAreaVehicleList.size()>0)

{

rwAreaVehicle = rwAreaVehicleList.get(0);

}

return rwAreaVehicle;

}

@Override

public Response getRWAreaVehiclePnuts(Order order)

{

Response resp = new Response();

try

{

IMESASOMRWAreaVehicle rwAreaVehicle = getRWAreaVehicle(order);

resp.setResult(rwAreaVehicle);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void inRWArea(Order order,String rwAreaName) throws DatasweepException, MESException

{

String functionName = "inRWArea(Order order,String rwAreaName)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

if(StringUtilsEx.isBlank(rwAreaName))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN rwAreaName is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "rwAreaName" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],rwAreaName:["+ rwAreaName + "];";

IMESASOMRWAreaVehicle rwAreaVehicle = getRWAreaVehicle(order);

if(rwAreaVehicle != null)

{

LogUtility.logError(

MODEL\_NAME, "Vehicles are in rework area! keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("Vehicles are in rework area! keyWord<" + keyWord + ">");

}

rwAreaVehicle = new MESASOMRWAreaVehicle();

rwAreaVehicle.setRwareaname(rwAreaName);

rwAreaVehicle.setOrder(order);

rwAreaVehicle.setIntime(PCContext.getFunctions().getDBTime());

Response response = rwAreaVehicle.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMRWAreaVehicle success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMRWAreaVehicle fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMRWAreaVehicle", keyWord });

}

saveRWAreaVehicleLog(order, rwAreaName,ReworkOperType.IN);

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response inRWAreaPnuts(Order order,String rwAreaName)

{

Response resp = new Response();

try

{

inRWArea(order, rwAreaName);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void outRwArea(Order order) throws MESException, DatasweepException

{

String functionName = "outRwArea(Order order)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "];";

IMESASOMRWAreaVehicle rwAreaVehicle = getRWAreaVehicle(order);

if(rwAreaVehicle == null)

{

LogUtility.logError(

MODEL\_NAME, "Vehicles are not in rework area! keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("Vehicles are not in! keyWord<" + keyWord + ">");

}

String rwAreaName = rwAreaVehicle.getRwareaname();

Response response = rwAreaVehicle.delete(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Delete MESASOMRWAreaVehicle success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Delete MESASOMRWAreaVehicle fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_DELETE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMRWAreaVehicle", keyWord });

}

saveRWAreaVehicleLog(order, rwAreaName,ReworkOperType.OUT);

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response outRwAreaPnuts(Order order)

{

Response resp = new Response();

try

{

outRwArea(order);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

private void saveRWAreaVehicleLog(Order order,String rwAreaName,Long operType) throws DatasweepException, MESException

{

String functionName = "saveRWAreaVehicleLog(Order order,String rwAreaName,Long orderType)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

if (StringUtilsEx.isBlank(rwAreaName))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN rwAreaName is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "rwAreaName" });

}

if (operType == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN operType is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "operType" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],rwAreaName:["+ rwAreaName + "],operType:["+ operType + "];";

IMESASOMRWAreaVehicleLog rwAreaVehicleLog = new MESASOMRWAreaVehicleLog();

rwAreaVehicleLog.setOrdernumber(order.getOrderNumber());

rwAreaVehicleLog.setBsn(OrderHelper.getBsn(order));

rwAreaVehicleLog.setMix(OrderHelper.getMIX(order));

rwAreaVehicleLog.setVin(OrderHelper.getVin(order));

rwAreaVehicleLog.setRwareaname(rwAreaName);

rwAreaVehicleLog.setOpertype(operType);

rwAreaVehicleLog.setOpertime(PCContext.getFunctions().getDBTime());

Response response= rwAreaVehicleLog.save(null,keyWord,null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMRWAreaVehicleLog success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMRWAreaVehicleLog fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMRWAreaVehicleLog", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public void calcBDCOutQueue() throws DatasweepException, MESException

{

String functionName = "calcBDCOutQueue()";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

Boolean isAutoCalc = getBDCOutQueueAutoCalc();

if(isAutoCalc)

{

Long calcNum = getBDCOutQueueCalcNumber();

Long maxRows = calcNum;

List<String> profileNameList = new ArrayList<String>();

Map<Long,IMESASOMBDCAttrGroup> bdcAttrGroupMap = new HashMap<Long,IMESASOMBDCAttrGroup>();

Map<Long, Map<String,String>> orderProfileMap = new HashMap<Long, Map<String,String>>();

MESASOMBDCStockFilter bdcStockFilter = new MESASOMBDCStockFilter();

bdcStockFilter.forIsinqueueEqualTo(false);

bdcStockFilter.forAreaEqualTo(BDCAreaName.PBS);

bdcStockFilter.forLocationNotEqualTo(null);

bdcStockFilter.addOrderBy(MESGeneratedASOMBDCStock.COL\_NAME\_SEQUENCETCF, IATRowFilterAttributes.ATCOLUMN,

IFilterSortOrders.ASCENDING);

List<IMESASOMBDCStock> bdcStockList = bdcStockFilter.getFilteredObjects();

IMESASOMBDCHoldVehicleFilter holdVehicleFilter = new MESASOMBDCHoldVehicleFilter();

holdVehicleFilter.forIsholdEqualTo(true);

List<IMESASOMBDCHoldVehicle> holdVehicleList = holdVehicleFilter.getFilteredObjects();

for(IMESASOMBDCHoldVehicle holdVehicle:holdVehicleList)

{

for(IMESASOMBDCStock bdcStock:bdcStockList)

{

if(holdVehicle.getOrder().getKey() == bdcStock.getOrder().getKey())

{

bdcStockList.remove(bdcStock);

}

}

}

IMESASOMBDCSpaceFilter bdcSpaceFilter = new MESASOMBDCSpaceFilter();

bdcSpaceFilter.forIsactiveEqualTo(true);

List<IMESASOMBDCSpace> bdcSpaceList = bdcSpaceFilter.getFilteredObjects();

for(IMESASOMBDCSpace bdcSpace:bdcSpaceList)

{

if(bdcSpace.getQty()>maxRows)

{

maxRows = bdcSpace.getQty();

}

bdcAttrGroupMap.put(bdcSpace.getBeforeattrgroupObj().getKey(), bdcSpace.getBeforeattrgroupObj());

bdcAttrGroupMap.put(bdcSpace.getAfterattrgroupObj().getKey(), bdcSpace.getAfterattrgroupObj());

}

IMESASOMBDCContinuousFilter bdcContinuousFilter = new MESASOMBDCContinuousFilter();

bdcContinuousFilter.forIsactiveEqualTo(true);

List<IMESASOMBDCContinuous> bdcContinuousList = bdcContinuousFilter.getFilteredObjects();

for(IMESASOMBDCContinuous bdcContinuous:bdcContinuousList)

{

if(bdcContinuous.getQty()>maxRows)

{

maxRows = bdcContinuous.getQty();

}

bdcAttrGroupMap.put(bdcContinuous.getAttrgroupObj().getKey(), bdcContinuous.getAttrgroupObj());

}

IMESASOMBDCRatioFilter bdcRatioFilter = new MESASOMBDCRatioFilter();

bdcRatioFilter.forIsactiveEqualTo(true);

List<IMESASOMBDCRatio> bdcRatioList = bdcRatioFilter.getFilteredObjects();

for(IMESASOMBDCRatio bdcRatio:bdcRatioList)

{

if(bdcRatio.getTotalqty()>maxRows)

{

maxRows = bdcRatio.getTotalqty();

}

bdcAttrGroupMap.put(bdcRatio.getAttrgroupObj().getKey(),bdcRatio.getAttrgroupObj());

}

MESASOMBDCOutQueueFilter bdcOutQueueFilter = new MESASOMBDCOutQueueFilter();

bdcOutQueueFilter.addOrderBy(MESGeneratedASOMBDCOutQueue.COL\_NAME\_SORTNO, IATRowFilterAttributes.ATCOLUMN,

IFilterSortOrders.DESCENDING);

bdcOutQueueFilter.setMaxRows(maxRows.intValue());

List<IMESASOMBDCOutQueue> bdcOutQueueList = bdcOutQueueFilter.getFilteredObjects();

for(IMESASOMBDCAttrGroup attrGroup : bdcAttrGroupMap.values())

{

List<MESASOMBDCAttrGroupItem> attrGroupItemList = attrGroup.getBDCAttrGroupItemList();

for(MESASOMBDCAttrGroupItem attrGroupItem:attrGroupItemList)

{

Boolean isFind = false;

String profile = attrGroupItem.getProfile();

for(String profileName:profileNameList)

{

if(profileName.equals(profile))

{

isFind = true;

break;

}

}

if(isFind == false)

{

profileNameList.add(profile);

}

}

}

for(IMESASOMBDCStock bdcStock:bdcStockList)

{

Order order = bdcStock.getOrder();

//TODO

Map<String,String> profileDataMap = getSystemServiceEx().getProfileDataMap(order, "BDC", profileNameList);

orderProfileMap.put(order.getKey(), profileDataMap);

if(bdcStock.getIsinqueue() == true)

{

calcNum = calcNum - 1;

}

}

for(IMESASOMBDCOutQueue bdcOutQueue:bdcOutQueueList)

{

Order order = bdcOutQueue.getOrder();

//TODO

Map<String,String> profileDataMap = getSystemServiceEx().getProfileDataMap(order, "BDC", profileNameList);

orderProfileMap.put(order.getKey(), profileDataMap);

}

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

{

for(IMESASOMBDCStock bdcStock:bdcStockList)

{

Boolean isOk = true;

Order order = bdcStock.getOrder();

for(IMESASOMBDCSpace bdcSpace:bdcSpaceList)

{

IMESASOMBDCAttrGroup attrGroup = bdcSpace.getAfterattrgroupObj();

if(isAccordAttrOfOrder(attrGroup, orderProfileMap.get(order.getKey())) == true)

{

Integer qty = bdcSpace.getQty().intValue();

Integer size = bdcOutQueueList.size();

Integer num = qty;

if(qty>size)

{

num = size;

}

for(int j = 0;j < num ; j++)

{

IMESASOMBDCOutQueue bdcOutQueue = bdcOutQueueList.get(j);

Order befterOrder = bdcOutQueue.getOrder();

IMESASOMBDCAttrGroup befterAttrGroup = bdcSpace.getBeforeattrgroupObj();

if(isAccordAttrOfOrder(befterAttrGroup, orderProfileMap.get(befterOrder.getKey())) == true)

{

isOk = false;

break;

}

}

if(isOk = false)

{

break;

}

}

}

if(isOk = false)

{

break;

}

for(IMESASOMBDCContinuous bdcContinuous:bdcContinuousList)

{

//TODO 确认是最小连续数，还是最大连续数，现在按最大连续数

Integer qty = bdcContinuous.getQty().intValue();

Integer size = bdcOutQueueList.size();

Integer num = qty;

if(qty>size)

{

isOk = false;

break;

}

IMESASOMBDCAttrGroup attrGroup = bdcContinuous.getAttrgroupObj();

if(isAccordAttrOfOrder(attrGroup, orderProfileMap.get(order.getKey())) == true)

{

for(int j = 0;j < num; j++)

{

IMESASOMBDCOutQueue bdcOutQueue = bdcOutQueueList.get(j);

Order befterOrder = bdcOutQueue.getOrder();

if(isAccordAttrOfOrder(attrGroup, orderProfileMap.get(befterOrder.getKey())) == false)

{

break;

}

}

//已经达到最大连续数,不符合规则

if(isOk = true)

{

isOk = false;

break;

}

}

}

if(isOk = false)

{

break;

}

for(IMESASOMBDCRatio bdcRatio:bdcRatioList)

{

Integer totalQty = bdcRatio.getTotalqty().intValue();

Integer ratioQty = bdcRatio.getRatioqty().intValue();

Integer size = bdcOutQueueList.size();

if(ratioQty>size)

{

continue;

}

Integer num = totalQty;

if(totalQty>size)

{

num = size;

}

Integer actualRatioQty = 0;

IMESASOMBDCAttrGroup attrGroup = bdcRatio.getAttrgroupObj();

if(isAccordAttrOfOrder(attrGroup, orderProfileMap.get(order.getKey())) == true)

{

for(int j = 0;j < num; j++)

{

IMESASOMBDCOutQueue bdcOutQueue = bdcOutQueueList.get(j);

Order befterOrder = bdcOutQueue.getOrder();

if(isAccordAttrOfOrder(attrGroup, orderProfileMap.get(befterOrder.getKey())) == true)

{

actualRatioQty++;

if(actualRatioQty>ratioQty)

{

isOk = false;

break;

}

}

}

if(isOk = true)

{

isOk = false;

break;

}

}

}

if(isOk = true)

{

IMESASOMBDCOutQueue bdcOutQueue = addOrderTobdcOutQueue(bdcStock, false);

bdcStockList.remove(bdcStock);

bdcOutQueueList.add(0, bdcOutQueue);

}

}

}

}

else

{

LogUtility.logInfo(MODEL\_NAME, "BDC manual calc out!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response calcBDCOutQueuePnuts()

{

Response resp = new Response();

try

{

calcBDCOutQueue();

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

private Long getBDCOutQueueCalcNumber()

{

Long calcNum = MESConfiguration.getMESConfiguration().getLong(BDC\_OUT\_QUEUE\_CALC\_NUMBER\_PROPERTY,

BDC\_OUT\_QUEUE\_CALC\_NUMBER, "");

return calcNum;

}

private Boolean getBDCOutQueueAutoCalc()

{

Boolean isAutonCalc = MESConfiguration.getMESConfiguration().getBoolean(BDC\_OUT\_QUEUE\_AUTO\_CALC\_PROPERTY,

true, "");

return isAutonCalc;

}

private Boolean isAccordAttrOfOrder(IMESASOMBDCAttrGroup attrGroup, Map<String,String> profileMap) throws MESException

{

String functionName = "isAccordAttrOfOrder(Order order, IMESASOMBDCAttrGroup attrGroup, Map<String,String> profileMap)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (attrGroup == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN attrGroup is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "attrGroup" });

}

if (profileMap == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN profileMap is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "profileMap" });

}

String keyWord = "attrGroupName["+attrGroup.getName()+"],";

Boolean result = true;

List<MESASOMBDCAttrGroupItem> attrGroupItemList = attrGroup.getBDCAttrGroupItemList();

for(MESASOMBDCAttrGroupItem attrGroupItem:attrGroupItemList)

{

String profile = attrGroupItem.getProfile();

String values = attrGroupItem.getValue();

String[] valueArray = values.split(",");

Long operType = attrGroupItem.getOperatorType();

String profileValue = profileMap.get(profile);

String keyWordItem = keyWord + "profile["+profile+"]";

//等于

if(operType == 10l)

{

result = false;

for(String value:valueArray)

{

if(profileValue.equals(value))

{

result = true;

break;

}

}

}

//不等于

else if(operType == 20l)

{

result = true;

for(String value:valueArray)

{

if(profileValue.equals(value))

{

result = false;

break;

}

}

}

else

{

LogUtility.logError(MODEL\_NAME,

"Operator Type[" + operType +"] is Error,keyWordItem<" + keyWordItem + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

if(result == false)

{

break;

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return result;

}

@Override

public IMESASOMBDCOutQueue addOrderTobdcOutQueue(IMESASOMBDCStock bdcStock, Boolean isManual) throws MESException

{

String functionName = "addOrderTobdcOutQueue(IMESASOMBDCStock bdcStock, Boolean isManual)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (bdcStock == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN bdcStock is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "bdcStock" });

}

if (isManual == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN isManual is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "isManual" });

}

Order order = bdcStock.getOrder();

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],isManual[" + isManual + "]";

bdcStock.setIsinqueue(true);

Response response = bdcStock.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMBDCStock success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMBDCStock fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMBDCStock", keyWord });

}

Long sortNo = getBDCOutQueueSortno();

IMESASOMBDCOutQueue bdcOutQueue = new MESASOMBDCOutQueue();

bdcOutQueue.setBsn(OrderHelper.getBsn(bdcStock.getOrder()));

bdcOutQueue.setIsmanual(isManual);

bdcOutQueue.setIsout(false);

bdcOutQueue.setOrder(order);

bdcOutQueue.setRequestcount(0l);

bdcOutQueue.setSortno(sortNo);

response = bdcOutQueue.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMBDCOutQueue success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

bdcStock.setIsinqueue(false);

response = bdcStock.save(null, keyWord, null);

LogUtility.logError(MODEL\_NAME,

"Save MESASOMBDCOutQueue fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMBDCOutQueue", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return bdcOutQueue;

}

@Override

public Response addOrderTobdcOutQueuePnuts(IMESASOMBDCStock bdcStock, Boolean isManual)

{

Response resp = new Response();

try

{

IMESASOMBDCOutQueue bdcOutQueue = addOrderTobdcOutQueue(bdcStock, isManual);

resp.setResult(bdcOutQueue);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void removeOrderFrombdcOutQueue(IMESASOMBDCOutQueue bdcOutQueue) throws DatasweepException, MESException

{

String functionName = "removeOrderFrombdcOutQueue(IMESASOMBDCOutQueue bdcOutQueue)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (bdcOutQueue == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN bdcOutQueue is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "bdcOutQueue" });

}

Order order = bdcOutQueue.getOrder();

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "]";

IMESASOMBDCStock bdcStock = getBDCStockObj(order,BDCAreaName.PBS);

if(bdcStock != null)

{

bdcStock.setIsinqueue(false);

Response response = bdcStock.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMBDCStock success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMBDCStock fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMBDCStock", keyWord });

}

}

Response response = bdcOutQueue.delete(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Delete MESASOMBDCOutQueue success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Delete MESASOMBDCOutQueue fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_DELETE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMBDCOutQueue", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response removeOrderFrombdcOutQueuePnuts(IMESASOMBDCOutQueue bdcOutQueue)

{

Response resp = new Response();

try

{

removeOrderFrombdcOutQueue(bdcOutQueue);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

private Long getBDCOutQueueSortno()

{

String functionName = "getBDCOutQueueSortno()";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

Long sortno = 0l;

MESASOMBDCOutQueueFilter bdcOutQueueFilter = new MESASOMBDCOutQueueFilter();

bdcOutQueueFilter.addOrderBy(MESGeneratedASOMBDCOutQueue.COL\_NAME\_SORTNO, IATRowFilterAttributes.ATCOLUMN,

IFilterSortOrders.DESCENDING);

bdcOutQueueFilter.setMaxRows(1);

List<IMESASOMBDCOutQueue> bdcOutQueueList = bdcOutQueueFilter.getFilteredObjects();

if(bdcOutQueueList.size()>0)

{

IMESASOMBDCOutQueue bdcOutQueue = bdcOutQueueList.get(0);

sortno = bdcOutQueue.getSortno();

}

sortno = sortno + 1;

LogUtility.logInfo(MODEL\_NAME, "get srotno is[" + sortno + "]",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return sortno;

}

@Override

public void generateOrSaveBuildDate(Order order) throws Exception

{

String functionName = "generateOrSaveBuildDate(Order order)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "];";

IMESASOMOrderProperty orderProperty = getSystemService().getOrGenerateOrderPropperty(order);

if(null == orderProperty)

{

LogUtility.logError(

MODEL\_NAME, "get orderProperty by order is null, keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("get orderProperty by order is null, keyWord<" + keyWord + ">");

}

Time buildDate = orderProperty.getBuilddate();

if(buildDate == null)

{

buildDate = generateBuildDate(order);

orderProperty.setBuilddate(buildDate);

Response responseOrderProperty = orderProperty.save(null,null,null);

if(responseOrderProperty.isError())

{

LogUtility.logError(MODEL\_NAME, "save orderProperty faild,Error Info["+responseOrderProperty.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order property faild, keyWord<" + keyWord + ">");

}

order.setUDA(buildDate, OrderUDAName.BUILD\_DATE);

Response responseOrder = order.save(null, keyWord, null);

if(responseOrder.isError())

{

LogUtility.logError(MODEL\_NAME, "save order faild,Error Info["+responseOrder.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

private Time generateBuildDate(Order order) throws DatasweepException, Exception

{

String buildDataYearStr = TimeHelper.timeToString((Time) order.getUDA(OrderUDAName.PLAN\_START\_DATE),"yyyy");

Integer buildDataYearInt = Integer.parseInt(buildDataYearStr);

Time currentTime = PCContext.getFunctions().getDBTime();

Integer dbTimeYear = currentTime.getYear();

Time buildTime = null;

if(buildDataYearInt > dbTimeYear)

{

buildTime = TimeHelper.stringToTime(buildDataYearInt+"-01-01 08:00:00","yyyy-MM-dd HH:mm:ss");

}

else if(buildDataYearInt < dbTimeYear)

{

buildTime = TimeHelper.stringToTime(buildDataYearInt+"-12-31 08:00:00","yyyy-MM-dd HH:mm:ss");

}

else

{

buildTime = currentTime;

}

return buildTime;

}

@Override

public Response generateOrSaveBuildDatePnuts(Order order)

{

Response resp = new Response();

try

{

generateOrSaveBuildDate(order);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void generateOrSaveBSN(Order order) throws Exception

{

String functionName = "generateOrSaveBSN(Order order)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "];";

IMESASOMOrderProperty orderProperty = getSystemService().getOrGenerateOrderPropperty(order);

if(null == orderProperty)

{

LogUtility.logError(

MODEL\_NAME, "get orderProperty by order is null, keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("get orderProperty by order is null, keyWord<" + keyWord + ">");

}

String bsn = orderProperty.getBsn();

if(StringUtilsEx.isEmpty(bsn))

{

bsn = generateBSN(order).toString();

orderProperty.setBsn(bsn);

Response responseOrderProperty = orderProperty.save(null,null,null);

if(responseOrderProperty.isError())

{

LogUtility.logError(MODEL\_NAME, "save orderProperty faild,Error Info["+responseOrderProperty.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

order.setUDA(bsn, OrderUDAName.BSN);

Response responseOrder = order.save(null, keyWord, null);

if(responseOrder.isError())

{

LogUtility.logError(MODEL\_NAME, "save order faild,Error Info["+responseOrder.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response generateOrSaveBSNPnuts(Order order)

{

Response resp = new Response();

try

{

generateOrSaveBSN(order);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void generateOrSaveVIN(Order order) throws Exception

{

String functionName = "generateOrSaveVIN(Order order)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "];";

IMESASOMOrderProperty orderProperty = getSystemService().getOrGenerateOrderPropperty(order);

if(null == orderProperty)

{

LogUtility.logError(

MODEL\_NAME, "get orderProperty by order is null, keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("get orderProperty by order is null, keyWord<" + keyWord + ">");

}

String vin = orderProperty.getVin();

if(StringUtilsEx.isEmpty(vin))

{

vin = generateVIN(order).toString();

orderProperty.setVin(vin);

Response response = orderProperty.save(null,null,null);

if(response.isError())

{

LogUtility.logError(MODEL\_NAME, "save orderProperty faild, keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

order.setUDA(vin, OrderUDAName.VIN);

Response responseOrder = order.save(null, keyWord, null);

if(responseOrder.isError())

{

LogUtility.logError(MODEL\_NAME, "save order faild,Error Info["+responseOrder.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response generateOrSaveVINPnuts(Order order)

{

Response resp = new Response();

try

{

generateOrSaveVIN(order);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void generateOrSaveMIX(Order order) throws Exception

{

String functionName = "generateOrSaveMIX(Order order)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "];";

IMESASOMOrderProperty orderProperty = getSystemService().getOrGenerateOrderPropperty(order);

if(null == orderProperty)

{

LogUtility.logError(

MODEL\_NAME, "get orderProperty by order is null, keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("get orderProperty by order is null, keyWord<" + keyWord + ">");

}

String mix = orderProperty.getMix();

if(StringUtilsEx.isEmpty(mix))

{

mix = generateMIX(order).toString();

orderProperty.setMix(mix);

Response response = orderProperty.save(null,null,null);

if(response.isError())

{

LogUtility.logError(MODEL\_NAME, "save orderProperty faild, keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response generateOrSaveMIXPnuts(Order order)

{

Response resp = new Response();

try

{

generateOrSaveMIX(order);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void generateOrSaveRFID(Order order, Long scenario) throws Exception

{

String functionName = " generateOrSaveRFID(Order order, Long scenario)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],scenario["+scenario+"];";

IMESASOMOrderProperty orderProperty = getSystemService().getOrGenerateOrderPropperty(order);

if(null == orderProperty)

{

LogUtility.logError(

MODEL\_NAME, "get orderProperty by order is null, keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("get orderProperty by order is null, keyWord<" + keyWord + ">");

}

String rfid = generateRFID(order, scenario).toString();

orderProperty.setRfid(rfid);

Response responseOrderProperty = orderProperty.save(null,keyWord,null);

if(responseOrderProperty.isError())

{

LogUtility.logError(MODEL\_NAME, "save orderProperty faild,Error Info["+responseOrderProperty.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

order.setUDA(rfid, OrderUDAName.RFID);

Response responseOrder = order.save(null, keyWord, null);

if(responseOrder.isError())

{

LogUtility.logError(MODEL\_NAME, "save order faild,Error Info["+responseOrder.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response generateOrSaveRFIDPnuts(Order order, Long scenario)

{

Response resp = new Response();

try

{

generateOrSaveRFID(order, scenario);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void updateShopSequence(Order order, Area shop,Long shopSequence) throws Exception

{

String functionName = "updateBiwShopSequence(Order order, Long shopSequence)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

if (shop == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN shop is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "shop" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],shopNmae["+shop.getName()+"],shopSequence["+shopSequence+"];";

IMESASOMOrderProperty orderProperty = getSystemService().getOrGenerateOrderPropperty(order);

if(null == orderProperty)

{

LogUtility.logError(

MODEL\_NAME, "get orderProperty by order is null, keyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("get orderProperty by order is null, keyWord<" + keyWord + ">");

}

orderProperty.setShopsequence(shopSequence);

Response responseOrderProperty = orderProperty.save(null,keyWord,null);

if(responseOrderProperty.isError())

{

LogUtility.logError(MODEL\_NAME, "save orderProperty faild,Error Info["+responseOrderProperty.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

OrderItem orderItem= OrderHelper.getOrderItemByShop(order, shop);

orderItem.setUDA(shopSequence, OrderItemUDAName.SHOP\_SEQUENCE);

Response responseOrder = order.save(null, keyWord, null);

if(responseOrder.isError())

{

LogUtility.logError(MODEL\_NAME, "save order faild,Error Info["+responseOrder.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response updateShopSequencePnuts(Order order,Area shop, Long shopSequence)

{

Response resp = new Response();

try

{

updateShopSequence(order, shop, shopSequence);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void bindingSkidCode(Order order, String skidCode, WorkCenter workCenter) throws DatasweepException, MESException

{

String functionName = "bindingSkidCode(Order order, String skidCode, WorkCenter workCenter)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN Order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "Order" });

}

if(StringUtilsEx.isEmpty(skidCode))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN skidCode is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "skidCode" });

}

if(workCenter == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN workCenter is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "workCenter" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "],skidCode["+skidCode+"]";

String bsn = OrderHelper.getBsn(order);

MESASOMSkidCodeBindingLogFilter bsnfilter = new MESASOMSkidCodeBindingLogFilter();

bsnfilter.forBsnEqualTo(bsn);

bsnfilter.forIsbindingEqualTo(true);

IMESASOMSkidCodeBindingLogFilter skidCodefilter = new MESASOMSkidCodeBindingLogFilter();

skidCodefilter.forSkidcodeEqualTo(skidCode);

skidCodefilter.forIsbindingEqualTo(true);

skidCodefilter.addOr(bsnfilter);

List<IMESASOMSkidCodeBindingLog> skidCodeBindingLogList = skidCodefilter.getFilteredObjects();

for(IMESASOMSkidCodeBindingLog skidCodeBindingLog:skidCodeBindingLogList)

{

skidCodeBindingLog.setIsbinding(false);

Response response = skidCodeBindingLog.save(null,keyWord,null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMSkidCodeBindingLog success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMSkidCodeBindingLog fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSkidCodeBindingLog", keyWord });

}

}

IMESASOMSkidCodeBindingLog skidCodeBindingLogObj = new MESASOMSkidCodeBindingLog();

skidCodeBindingLogObj.setBindingtime(PCContext.getFunctions().getDBTime());

skidCodeBindingLogObj.setBsn(bsn);

skidCodeBindingLogObj.setLocation(workCenter.getName());

skidCodeBindingLogObj.setSkidcode(skidCode);

skidCodeBindingLogObj.setUser(PCContext.getFunctions().getCurrentUser().getName());

skidCodeBindingLogObj.setIsbinding(true);

Response response = skidCodeBindingLogObj.save(null,keyWord,null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMSkidCodeBindingLog success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMSkidCodeBindingLog fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSkidCodeBindingLog", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response bindingSkidCodePnuts(Order order, String skidCode, WorkCenter workCenter)

{

Response resp = new Response();

try

{

bindingSkidCode(order, skidCode, workCenter);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void clearSkidCode(Order order) throws DatasweepException, MESException

{

String functionName = "clearSkidCode(Order order)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN Order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "Order" });

}

String keyWord = "OrderNumber:[" + order.getOrderNumber() + "]";

String bsn = OrderHelper.getBsn(order);

MESASOMSkidCodeBindingLogFilter skidCodefilter = new MESASOMSkidCodeBindingLogFilter();

skidCodefilter.forBsnEqualTo(bsn);

skidCodefilter.forIsbindingEqualTo(true);

List<IMESASOMSkidCodeBindingLog> skidCodeBindingLogList = skidCodefilter.getFilteredObjects();

for(IMESASOMSkidCodeBindingLog skidCodeBindingLog:skidCodeBindingLogList)

{

skidCodeBindingLog.setIsbinding(false);

Response response = skidCodeBindingLog.save(null,keyWord,null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMSkidCodeBindingLog success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMSkidCodeBindingLog fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSkidCodeBindingLog", keyWord });

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response clearSkidCodePnuts(Order order)

{

Response resp = new Response();

try

{

clearSkidCode(order);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void rebindingSkidCode(Order oldOrder, Order newOrder, WorkCenter workCenter) throws DatasweepException, MESException

{

String functionName = "rebindingSkidCode(Order oldOrder, Order newOrder, WorkCenter workCenter)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(oldOrder == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN oldOrder is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "Order" });

}

if(newOrder == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN newOrder is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "Order" });

}

if(workCenter == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN workCenter is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "workCenter" });

}

String oldBsn = OrderHelper.getBsn(oldOrder);

String newBsn = OrderHelper.getBsn(newOrder);

String keyWord = "oldBsn:[" + oldBsn + "],newBsn:[" + newBsn + "],wcName["+workCenter.getName()+"]";

LogUtility.logInfo(

MODEL\_NAME, "Function Parameter values,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

String skidCode = "";

MESASOMSkidCodeBindingLogFilter oldSkidCodeBindingLogFilter = new MESASOMSkidCodeBindingLogFilter();

oldSkidCodeBindingLogFilter.forBsnEqualTo(oldBsn);

oldSkidCodeBindingLogFilter.addOrderBy(IATRowFilterAttributes.CREATIONTIME, IFilterSortOrders.DESCENDING);

oldSkidCodeBindingLogFilter.setMaxRows(1);

if(oldSkidCodeBindingLogFilter.getCount()>0)

{

IMESASOMSkidCodeBindingLog skidCodeBindingLog = oldSkidCodeBindingLogFilter.getFilteredObjects().get(0);

skidCode = skidCodeBindingLog.getSkidcode();

LogUtility.logInfo(

MODEL\_NAME, "get skidCode[" + skidCode + "],KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"get skidCode is null,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("get skidCode is null,KeyWord<" + keyWord + ">");

}

MESASOMSkidCodeBindingLogFilter bsnfilter = new MESASOMSkidCodeBindingLogFilter();

bsnfilter.forBsnEqualTo(oldBsn);

bsnfilter.forBsnEqualTo(newBsn);

bsnfilter.forIsbindingEqualTo(true);

IMESASOMSkidCodeBindingLogFilter skidCodefilter = new MESASOMSkidCodeBindingLogFilter();

skidCodefilter.forSkidcodeEqualTo(skidCode);

skidCodefilter.forIsbindingEqualTo(true);

skidCodefilter.addOr(bsnfilter);

List<IMESASOMSkidCodeBindingLog> skidCodeBindingLogList = skidCodefilter.getFilteredObjects();

for(IMESASOMSkidCodeBindingLog skidCodeBindingLog:skidCodeBindingLogList)

{

skidCodeBindingLog.setIsbinding(false);

Response response = skidCodeBindingLog.save(null,keyWord,null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "update MESASOMSkidCodeBindingLog success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

LogUtility.logError(MODEL\_NAME,

"update MESASOMSkidCodeBindingLog fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSkidCodeBindingLog", keyWord });

}

}

IMESASOMSkidCodeBindingLog skidCodeBindingLogObj = new MESASOMSkidCodeBindingLog();

skidCodeBindingLogObj.setBindingtime(PCContext.getFunctions().getDBTime());

skidCodeBindingLogObj.setBsn(newBsn);

skidCodeBindingLogObj.setLocation(workCenter.getName());

skidCodeBindingLogObj.setSkidcode(skidCode);

skidCodeBindingLogObj.setUser(PCContext.getFunctions().getCurrentUser().getName());

skidCodeBindingLogObj.setIsbinding(true);

Response response = skidCodeBindingLogObj.save(null,keyWord,null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMSkidCodeBindingLog success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMSkidCodeBindingLog fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSkidCodeBindingLog", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response rebindingSkidCodePnuts(Order oldOrder, Order newOrder, WorkCenter workCenter)

{

Response resp = new Response();

try

{

rebindingSkidCode(oldOrder, newOrder, workCenter);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public String getRfidByEpc(String epc) throws DatasweepException, MESException

{

String functionName = "getRfidByEpc(String epc)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(StringUtilsEx.isEmpty(epc))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN epc is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "epc" });

}

String keyWord = "epc:[" + epc + "]";

String rfid = "";

IMESASOMRFIDAndEPCBindingFilter rfidAndEPCBindingFilter = new MESASOMRFIDAndEPCBindingFilter();

rfidAndEPCBindingFilter.forEpcnumberEqualTo(epc);

List<IMESASOMRFIDAndEPCBinding> vectorRfidAndEPCBinding = rfidAndEPCBindingFilter.getFilteredObjects();

if(vectorRfidAndEPCBinding.size() > 0)

{

IMESASOMRFIDAndEPCBinding rfidAndEPCBinding = vectorRfidAndEPCBinding.get(0);

Order order = rfidAndEPCBinding.getWorkorder();

rfid = OrderHelperEx.getRFID(order);

}

if(StringUtilsEx.isEmpty(rfid))

{

LogUtility.logError(

MODEL\_NAME, " get rfid is null! KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "rfid" });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return rfid;

}

@Override

public Response getRfidByEpcPnuts(String epc)

{

Response resp = new Response();

try

{

resp.setResult(getRfidByEpc(epc));

}

catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void steelBarcodeBinding(String bsn, String orderType,Station station,String steelCode) throws MESException, DatasweepException

{

String functionName = "steelBarcodeBinding(String bsn, String orderType,Station station,String steelCode)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(StringUtilsEx.isEmpty(bsn))

{

LogUtility.logError(

MODEL\_NAME, " bsn is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { " bsn" });

}

if(StringUtilsEx.isEmpty(orderType))

{

LogUtility.logError(

MODEL\_NAME, "orderType is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderType" });

}

if(station == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN station is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "station" });

}

if(StringUtilsEx.isEmpty(steelCode))

{

LogUtility.logError(

MODEL\_NAME, "steelCode is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "steelCode" });

}

String keyWord = "bsn:[" + bsn + "],orderType["+orderType+"],stationName["+station.getName()+"],steelCode["+steelCode+"]";

Time curTime = PCContext.getFunctions().getDBTime();

IMESASOMSteelCodeBindingFilter steelCodeBindingFilter = new MESASOMSteelCodeBindingFilter();

steelCodeBindingFilter.forBsnEqualTo(bsn);

steelCodeBindingFilter.forOrdertypeEqualTo(orderType);

List<IMESASOMSteelCodeBinding> steelCodeBindingList = steelCodeBindingFilter.getFilteredObjects();

IMESASOMSteelCodeBinding steelCodeBinding = null;

if(steelCodeBindingList.size()>0)

{

steelCodeBinding = steelCodeBindingList.get(0);

}

else

{

steelCodeBinding = new MESASOMSteelCodeBinding();

steelCodeBinding.setBsn(bsn);

steelCodeBinding.setOrdertype(orderType);

steelCodeBinding.setSteelcode(steelCode);

}

steelCodeBinding.setOperationstation(station);

steelCodeBinding.setOperationtime(curTime);

steelCodeBinding.setOperationuser(PCContext.getFunctions().getCurrentUser().getName());

Response responseSteelCodeBinding = steelCodeBinding.save(null, keyWord, null);

if (responseSteelCodeBinding.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMSteelCodeBinding success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMSteelCodeBinding fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ responseSteelCodeBinding.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSteelCodeBinding", keyWord });

}

IMESASOMSteelCodeBindingLog steelCodeBindingLog = new MESASOMSteelCodeBindingLog();

steelCodeBindingLog.setBsn(bsn);

steelCodeBindingLog.setOrdertype(orderType);

steelCodeBindingLog.setOperationstation(station);

steelCodeBindingLog.setSteelcode(steelCode);

steelCodeBindingLog.setOperationtime(curTime);

steelCodeBindingLog.setOperationuser(PCContext.getFunctions().getCurrentUser().getName());

Response responseSteelCodeBindingLog = steelCodeBindingLog.save(null, keyWord, null);

if (responseSteelCodeBindingLog.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMSteelCodeBindingLog success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMSteelCodeBindingLog fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ responseSteelCodeBindingLog.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSteelCodeBindingLog", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response steelBarcodeBindingPnuts(String bsn, String orderType,Station station,String steelCode)

{

Response resp = new Response();

try

{

steelBarcodeBinding(bsn,orderType,station,steelCode);

}

catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void updateAllUnitOfVINByOrder(Order order) throws DatasweepException, MESException

{

String functionName = "updateAllUnitOfVIN(Order order,String vin)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

String vin = OrderHelper.getVin(order);

String keyWord = "orderNumber:[" + order.getOrderNumber() + "],vin["+vin+"]";

Vector<Unit> unitList = order.getAllUnits();

for(Unit unit:unitList)

{

unit.setUDA(vin, UnitUDAName.VIN);

Response responseUnit = unit.save(null, keyWord, null);

if(responseUnit.isError())

{

LogUtility.logError(MODEL\_NAME, "save unit faild,Error Info["+responseUnit.getFirstErrorMessage()+"],unitKey["+unit.getKey()+"] keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save unit faild, keyWord<" + keyWord + ">");

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response updateAllUnitOfVINByOrderPnuts(Order order)

{

Response resp = new Response();

try

{

updateAllUnitOfVINByOrder(order);

}

catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void updateAllUnitOfRFIDByOrder(Order order) throws DatasweepException, MESException

{

String functionName = "updateAllUnitOfRFID(Order order,String rfid)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

String rfid = OrderHelper.getRFID(order);

String keyWord = "orderNumber:[" + order.getOrderNumber() + "],rfid["+rfid+"]";

Vector<Unit> unitList = order.getAllUnits();

for(Unit unit:unitList)

{

unit.setUDA(rfid, UnitUDAName.RFID);

Response responseUnit = unit.save(null, keyWord, null);

if(responseUnit.isError())

{

LogUtility.logError(MODEL\_NAME, "save unit faild,Error Info["+responseUnit.getFirstErrorMessage()+"],unitKey["+unit.getKey()+"] keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save unit faild, keyWord<" + keyWord + ">");

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response updateAllUnitOfRFIDByOrderPnuts(Order order)

{

Response resp = new Response();

try

{

updateAllUnitOfRFIDByOrder(order);

}

catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public MESASSMCalendar getLastCalendar(String category, String targetType, String target,

Time time)

throws DatasweepException

{

MESASSMCalendar calendar = null;

MESASSMCalendarFilter filter = new MESASSMCalendarFilter();

filter.forCategoryEqualTo(

category);

filter.forTargettypeEqualTo(

targetType);

filter.forTargetEqualTo(

target);

filter.forStarttimeLessThan(

time);

filter.addOrderATColumnBy(

"start\_time", IFilterSortOrders.DESCENDING);

filter.setMaxRows(

1);

List<IMESASSMCalendar> list = filter.getFilteredObjects();

// Vector<MESASSMCalendar> vector = filter.exec();

if (list.size() > 0)

{

calendar = (MESASSMCalendar) list.get(

0);

}

return calendar;

}

@Override

public Response getLastCalendarPnuts(String category, String targetType, String target,

Time time)

{

Response resp = new Response();

try

{

resp.setResult(

getLastCalendar(

category, targetType, target, time));

}

catch (Exception e)

{

resp.addError(

new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public IMESASOMPlanProduction getOrderPlanProductionObj(String orderNumber)

throws DatasweepException

{

MESASOMPlanProductionFilter filter = new MESASOMPlanProductionFilter();

filter.forOrdernumberEqualTo(

orderNumber);

filter.setMaxRows(

1);

List<IMESASOMPlanProduction> planProductionList = filter.getFilteredObjects();

IMESASOMPlanProduction mPlanProduction = null;

if (planProductionList.size() > 0)

{

mPlanProduction = planProductionList.get(

0);

}

return mPlanProduction;

}

@Override

public Response getOrderPlanProductionObjPnuts(String orderNumber)

{

Response resp = new Response();

try

{

resp.setResult(

getOrderPlanProductionObj(orderNumber));

}

catch (Exception e)

{

resp.addError(

new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void bindingSubPart(Order order,String orderType,String steelCode)

throws MESException,

DatasweepException

{

String functionName = "bindingSubPart(Order order,String orderType,String steelCode)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(order == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN order is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "order" });

}

if(StringUtilsEx.isEmpty(orderType))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN orderType is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderType" });

}

if(StringUtilsEx.isEmpty(steelCode))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN steelCode is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "steelCode" });

}

String keyWord = "orderNumber:[" + order.getOrderNumber() + "],orderType["+orderType+"],steelCode["+steelCode+"]";

IMESASOMSteelCodeBindingFilter steelCodeBindingFilter = new MESASOMSteelCodeBindingFilter();

steelCodeBindingFilter.forOrdertypeEqualTo(orderType);

steelCodeBindingFilter.forSteelcodeEqualTo(steelCode);

Long steelCodeBindingCount = steelCodeBindingFilter.getCount();

if(steelCodeBindingCount<1)

{

LogUtility.logError(

IModuleName.MODULE\_OM,

"steelCode not found, KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_FAIL\_TO\_GET\_DATA, new Object[] { "steelCode" });

}

IMESASOMSubPartBindingFilter subPartBindingFilter = new MESASOMSubPartBindingFilter();

subPartBindingFilter.forOrdertypeEqualTo(orderType);

subPartBindingFilter.forOrderEqualTo(order);

IMESASOMSubPartBinding subPartBindingObj = null;

if(subPartBindingFilter.getCount()>0)

{

subPartBindingObj =subPartBindingFilter.getFilteredObjects().get(0);

}

else

{

subPartBindingObj = new MESASOMSubPartBinding();

subPartBindingObj.setOrder(order);

subPartBindingObj.setOrdertype(orderType);

}

subPartBindingObj.setSteelcode(steelCode);

Response response = subPartBindingObj.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMSubPartBinding success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMSubPartBinding fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSubPartBinding", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response bindingSubPartPnuts(Order order,String orderType,String steelCode)

{

Response resp = new Response();

try

{

bindingSubPart(order, orderType, steelCode);

}

catch (Exception e)

{

resp.addError(

new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public Boolean updateSQD(String stationName,Time passTime) throws DatasweepException, MESException

{

String functionName = "updateSQD(String stationName,Time passTime)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(StringUtilsEx.isEmpty(stationName))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN orderType is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderType" });

}

if(passTime == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN steelCode is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "steelCode" });

}

String keyWord = "stationName:[" +stationName + "],passTime["+passTime+"]";

Boolean result = false;

MESASOMSQDFilter filter = new MESASOMSQDFilter();

filter.forStationnameEqualTo(stationName);

filter.forStatusEqualTo("0");

filter.addOrderBy(IATRowFilterAttributes.CREATIONTIME, IFilterSortOrders.ASCENDING);

filter.setMaxRows(1);

if(filter.getCount() > 0L)

{

List<IMESASOMSQD> sdqDataVector = filter.getFilteredObjects();

IMESASOMSQD sqdData = (IMESASOMSQD)sdqDataVector.get(0);

keyWord = keyWord + ",bsn[" + sqdData.getBsn() + "]";

sqdData.setPasstime(passTime);

sqdData.setStatus("1");

Response response = sqdData.save(null, keyWord, null);

if (response.isOk())

{

result = true;

LogUtility.logInfo(MODEL\_NAME, "Update sqdData success, KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Update sqdData fail,, KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSQD", keyWord });

}

}

else

{

LogUtility.logInfo(MODEL\_NAME, "get sdqdata is null, KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return result;

}

@Override

public Response updateSQDPnuts(String stationName,Time passTime)

{

Response resp = new Response();

try

{

resp.setResult(updateSQD(stationName, passTime));

}

catch (Exception e)

{

resp.addError(

new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public String getSteelCodeByBsn(String bsn,String orderType) throws DatasweepException

{

String steelCode = null;

IMESASOMSteelCodeBindingFilter steelCodeBindingFilter = new MESASOMSteelCodeBindingFilter();

steelCodeBindingFilter.forBsnEqualTo(bsn);

steelCodeBindingFilter.forOrdertypeEqualTo(orderType);

List<IMESASOMSteelCodeBinding> steelCodeBindingList = steelCodeBindingFilter.getFilteredObjects();

if(steelCodeBindingList.size()>0)

{

IMESASOMSteelCodeBinding steelCodeBinding = steelCodeBindingList.get(0);

steelCode = steelCodeBinding.getSteelcode();

}

return steelCode;

}

@Override

public Response getSteelCodeByBsnPnuts(String bsn,String orderType)

{

Response resp = new Response();

try

{

resp.setResult(getSteelCodeByBsn(bsn, orderType));

}

catch (Exception e)

{

resp.addError(

new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void syncTakeInOutStatus(String vin,Long status,WorkCenter workCenter,Long operationSource,String remark) throws DatasweepException, MESException

{

String functionName = "syncTakeInOutStatus(String vin,Long status,WorkCenter workCenter,Long operationSource,String remark)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(StringUtilsEx.isEmpty(vin))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN vin is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "vin" });

}

if(status == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN status is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "status" });

}

if(workCenter == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN workCenter is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "workCenter" });

}

if(operationSource == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN operationSource is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "operationSource" });

}

String keyWord = "vin:[" +vin + "],status:[" + status + "],workCenter:[" +workCenter.getName() + "],operationSource["+operationSource+"]";

IMESASOMSyncTakeInOutStatusFilter syncTakeInOutStatusFilter = new MESASOMSyncTakeInOutStatusFilter();

syncTakeInOutStatusFilter.forVinEqualTo(vin);

List<IMESASOMSyncTakeInOutStatus> syncTakeInOutStatusList = syncTakeInOutStatusFilter.getFilteredObjects();

IMESASOMSyncTakeInOutStatus syncTakeInOutStatusObj = null;

if(syncTakeInOutStatusList.size()>0)

{

syncTakeInOutStatusObj = syncTakeInOutStatusList.get(0);

}

else

{

syncTakeInOutStatusObj = new MESASOMSyncTakeInOutStatus();

syncTakeInOutStatusObj.setVin(vin);

}

syncTakeInOutStatusObj.setOperationstation(workCenter);

syncTakeInOutStatusObj.setOperationuser(PCContext.getFunctions().getCurrentUser().getName());

syncTakeInOutStatusObj.setOperationdate(PCContext.getFunctions().getDBTime());

syncTakeInOutStatusObj.setOperationsource(operationSource);

syncTakeInOutStatusObj.setStatus(status);

syncTakeInOutStatusObj.setRemark(remark);

Response response = syncTakeInOutStatusObj.save(null, vin, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMSyncTakeInOutStatus success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMSyncTakeInOutStatus fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMSyncTakeInOutStatus", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response syncTakeInOutStatusPnuts(String vin,Long status,WorkCenter workCenter,Long operationSource,String remark)

{

Response resp = new Response();

try

{

syncTakeInOutStatus(vin, status, workCenter, operationSource,remark);

}

catch (Exception e)

{

resp.addError(

new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void takeInOutBarcodeBind(String vin,String barcode,String orderType,WorkCenter workCenter) throws MESException, DatasweepException

{

String functionName = "takeInOutBarcodeBind(String vin,String barcode,Long orderType,WorkCenter workCenter)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(StringUtilsEx.isEmpty(vin))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN vin is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "vin" });

}

if(StringUtilsEx.isEmpty(barcode))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN barcode is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "barcode" });

}

if(StringUtilsEx.isEmpty(orderType))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN orderType is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderType" });

}

if(workCenter == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN workCenter is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "workCenter" });

}

String keyWord = "vin:[" +vin + "],barcode:[" + barcode + "],orderType:[" + orderType + "],workCenter:[" +workCenter.getName() + "]";

IMESASOMTakeInOutBCodeBind takeInOutBCodeBindObj = null;

IMESASOMTakeInOutBCodeBindFilter takeInOutBCodeBindFilter = new MESASOMTakeInOutBCodeBindFilter();

takeInOutBCodeBindFilter.forBarcodeEqualTo(barcode);

takeInOutBCodeBindFilter.forOrdertypeEqualTo(orderType);

List<IMESASOMTakeInOutBCodeBind> takeInOutBCodeBindList = takeInOutBCodeBindFilter.getFilteredObjects();

if(takeInOutBCodeBindList.size()>0)

{

takeInOutBCodeBindObj = takeInOutBCodeBindList.get(0);

}

else

{

takeInOutBCodeBindObj = new MESASOMTakeInOutBCodeBind();

}

takeInOutBCodeBindObj.setVin(vin);

takeInOutBCodeBindObj.setBarcode(barcode);

takeInOutBCodeBindObj.setOrdertype(orderType);

takeInOutBCodeBindObj.setOperationstation(workCenter);

takeInOutBCodeBindObj.setOperationdate(PCContext.getFunctions().getDBTime());

Response response = takeInOutBCodeBindObj.save(null, vin, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMTakeInOutBCodeBind success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMTakeInOutBCodeBind fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMTakeInOutBCodeBind", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response takeInOutBarcodeBindPnuts(String vin,String barcode,String orderType,WorkCenter workCenter)

{

Response resp = new Response();

try

{

takeInOutBarcodeBind(vin, barcode, orderType, workCenter);

}

catch (Exception e)

{

resp.addError(

new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public Boolean isSametakeInOutBarcode(String vin,String barcode,String orderType) throws MESException, DatasweepException

{

String functionName = "isSametakeInOutBarcode(String vin,String barcode,String orderType)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(StringUtilsEx.isEmpty(vin))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN vin is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "vin" });

}

if(StringUtilsEx.isEmpty(barcode))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN barcode is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "barcode" });

}

if(StringUtilsEx.isEmpty(orderType))

{

LogUtility.logError(

MODEL\_NAME, "GIVEN orderType is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "orderType" });

}

Boolean isSame = false;

IMESASOMTakeInOutBCodeBindFilter takeInOutBCodeBindFilter = new MESASOMTakeInOutBCodeBindFilter();

takeInOutBCodeBindFilter.forVinEqualTo(vin);

takeInOutBCodeBindFilter.forBarcodeEqualTo(barcode);

takeInOutBCodeBindFilter.forOrdertypeEqualTo(orderType);

if(takeInOutBCodeBindFilter.getCount()>0)

{

isSame = true;

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return isSame;

}

@Override

public Response isSametakeInOutBarcodePnuts(String vin,String barcode,String orderType)

{

Response resp = new Response();

try

{

resp.setResult(isSametakeInOutBarcode(vin, barcode, orderType));

}

catch (Exception e)

{

resp.addError(

new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public void generateBatchSubOrderByBIWSubOrder(Long quantity) throws Exception

{

String functionName = " generateBatchSubOrderByBIWSubOrder()";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

//获取各分线类型

String typeSql = "select \n" +

" NAME\_S \n" +

"from AT\_AS\_SM\_SUBLINETYPE \n" +

"where TYPE\_I = " + SubLineType.BATCH;

Vector typeVector = PCContext.getFunctions().getArrayDataFromActive(typeSql);

if (typeVector.size() > 0)

{

String subLineTypeSql = "";

for (int t = 0; t < typeVector.size(); t ++)

{

String[] type = (String[]) typeVector.elementAt(t);

String name = type[0];

subLineTypeSql += "select \n" +

" SUB\_LINE\_TYPE\_S,\n" +

" PART\_NO\_S \n" +

"from \n" +

"(select \n" +

" SUB\_LINE\_TYPE\_S,\n" +

" PART\_NO\_S \n" +

"from AT\_AS\_OM\_BATCHSUBORDER \n" +

"where SUB\_LINE\_TYPE\_S = '"+name+"' \n" +

"order by SEQUENCE\_S desc ) \n" +

"where rownum = 1 \n" +

"UNION ";

}

//-获取各分线最晚生产订单的物料号

Vector subLineTypeVector = null;

if (subLineTypeSql.length() > 0) {

subLineTypeSql = subLineTypeSql.substring(0, subLineTypeSql.length()-6);

subLineTypeVector = PCContext.getFunctions().getArrayDataFromActive(subLineTypeSql);

}

//获取需要更新状态的子工单key

String subOrderKeySql = "select \n" +

" biworder.ORDER\_NUMBER,\n" +

" biworder.ATR\_KEY \n" +

"from \n" +

"(select \n" +

" bso.ATR\_KEY,\n" +

" bso.SEQUENCE\_S,\n" +

" wo.ORDER\_NUMBER \n" +

"from AT\_AS\_OM\_BIW\_SUBORDER bso \n" +

"left join UNIT u on bso.SHOP\_ORDER\_47 = u.UNIT\_KEY \n" +

"left join UDA\_UNIT uu on u.UNIT\_KEY = uu.OBJECT\_KEY \n" +

"left join WORK\_ORDER wo on u.ORDER\_KEY = wo.ORDER\_KEY \n" +

"where bso.ZONE\_S = '"+OrderZone.CBL+"' \n" +

"and bso.IS\_BROADCASTED\_Y = 0 \n" +

"order by bso.SEQUENCE\_S ) biworder \n" +

"where rownum <= "+quantity;

Vector subOrderKeyVectory = PCContext.getFunctions().getArrayDataFromActive(subOrderKeySql);

//分类四门两盖顶盖数量

String totalSql = "select \n" +

" pi.SUB\_LINE\_TYPE\_S,\n" +

" so.SEQUENCE\_S,\n" +

" pi.PART\_NO\_S,\n" +

" pi.JOB\_NO\_I,\n" +

" pi.PACK\_QTY\_I,\n" +

" so.EIG\_PLANT\_S \n" +

//"from AT\_AS\_OM\_ORDER\_BOM ob \n" +

//"left join PART p on ob.PART\_21 = p.PART\_KEY \n" +

"from at\_as\_om\_orderproperty op \n" +

"left join at\_as\_om\_jmc\_orderitem joi on op.vin\_s=joi.vinno\_s \n" +

"left join at\_as\_om\_bodybom bb on joi.hz\_body\_no\_s=bb.hz\_body\_no\_s \n" +

"left join AT\_AS\_OM\_BIWPARTINFO pi on bb.partno\_s = pi.part\_no\_s \n" + //p.PART\_NUMBER = pi.PART\_NO\_S

"left join \n" +

"(select \n" +

" biworder.ORDER\_KEY,\n" +

" biworder.ORDER\_NUMBER,\n" +

" biworder.SEQUENCE\_S,\n" +

" biworder.EIG\_PLANT\_S \n" +

"from \n" +

"(select \n" +

" wo.ORDER\_KEY,\n" +

" wo.ORDER\_NUMBER,\n" +

" bso.SEQUENCE\_S,\n" +

" uo.EIG\_PLANT\_S \n" +

"from AT\_AS\_OM\_BIW\_SUBORDER bso \n" +

"left join UNIT u on bso.SHOP\_ORDER\_47 = u.UNIT\_KEY \n" +

"left join UDA\_UNIT uu on u.UNIT\_KEY = uu.OBJECT\_KEY \n" +

"left join WORK\_ORDER wo on u.ORDER\_KEY = wo.ORDER\_KEY \n" +

"left join UDA\_ORDER uo on uo.OBJECT\_KEY = wo.ORDER\_KEY \n" +

"where bso.ZONE\_S = '"+OrderZone.CBL+"' \n" +

"and bso.IS\_BROADCASTED\_Y = 0 \n" +

"order by bso.SEQUENCE\_S ) biworder \n" +

"where rownum <= "+quantity+") so on so.ORDER\_KEY = op.order\_54 \n" + //ob.WORK\_ORDER\_I

"where so.ORDER\_KEY is not null \n" +

"and pi.SUB\_LINE\_TYPE\_S in \n" +

"(\n" +

"select \n" +

" NAME\_S \n" +

"from AT\_AS\_SM\_SUBLINETYPE \n" +

"where TYPE\_I = "+SubLineType.BATCH+" \n" +

")\n" +

"order by \n" +

" pi.SUB\_LINE\_TYPE\_S,\n" +

" so.SEQUENCE\_S";

Vector totalVector = PCContext.getFunctions().getArrayDataFromActive(totalSql);

// String keyWord = "";

//保证各分线连贯生产

if (subLineTypeVector != null && subLineTypeVector.size() > 0)

{

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

{

String[] subLineType = (String[]) subLineTypeVector.elementAt(i);

String type = subLineType[0];

String partNo = subLineType[1];

Long count = 0L;

for (int j = 0; j < totalVector.size(); j ++)

{

String[] subLineOrder = (String[]) totalVector.elementAt(j);

String type1 = subLineOrder[0];

String partNo1 = subLineOrder[2];

if (type.equals(type1) && partNo.equals(partNo1))

{

// String sequence1 = subLineOrder[1];

Long jobNo1 = Long.valueOf(subLineOrder[3]);

Long packQty1 = Long.valueOf(subLineOrder[4]);

String plantNo1 = subLineOrder[5];

count ++;

//统计数量等于包装量 或 最后一笔数据 或 当前物料号与下一个物料号不同

if (count == packQty1 || j+1 == totalVector.size() || !(((String[]) totalVector.elementAt(j+1))[2]).equals(partNo1))

{

savaMESASOMBatchSubOrder(type1,plantNo1,partNo1,count,jobNo1);

count = 0L;

totalVector.remove(j);

j--;

continue;

}

totalVector.remove(j);

j--;

}

}

}

}

//创建各分线剩余订单

Long count = 0L;

String partNo = "";

for (int k = 0; k < totalVector.size(); k ++)

{

String[] subLineOrder = (String[]) totalVector.elementAt(k);

String type1 = subLineOrder[0];

String partNo1 = subLineOrder[2];

Long jobNo1 = Long.valueOf(subLineOrder[3]);

Long packQty1 = Long.valueOf(subLineOrder[4]);

String plantNo1 = subLineOrder[5];

count ++;

//统计数量等于包装量 或 最后一笔数据 或 当前物料号与下一个物料号不同

if (count == packQty1 || k+1 == totalVector.size() || !(((String[]) totalVector.elementAt(k+1))[2]).equals(partNo1))

{

savaMESASOMBatchSubOrder(type1,plantNo1,partNo1,count,jobNo1);

count = 0L;

}

}

//更新订单状态

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

{

String[] subOrderKey = (String[]) subOrderKeyVectory.elementAt(i);

// String orderNo = subOrderKey[0];

String key = subOrderKey[1];

updateMESASOMBIWSubOrder(Long.valueOf(key));

}

}

else

{

LogUtility.logInfo(MODEL\_NAME, "No find subLineType",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response generateBatchSubOrderByBIWSubOrderPnuts(Long quantity)

{

Response resp = new Response();

try

{

generateBatchSubOrderByBIWSubOrder(quantity);

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

public Response savaMESASOMBatchSubOrder(String subLineType, String plantNo, String partNo, Long count, Long jobNo)

{

String functionName = " savaMESASOMBatchSubOrder(String[] subLineOrder,UserSequence userSequence) ";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

String keyWord = "subLineType:[" +subLineType + "],plantNo:[" +plantNo + "],partNo:[" +partNo + "],count:[" +count + "],jobNo:[" +jobNo + "]";

Response resp = new Response();

try

{

//通过分线类型获取排序号

String sequence = getSubSequenceBySubLineType(subLineType);

//通过分线类型和年份获取工单号

String OrderNo = getOrderNoBySubLineType(subLineType);

IMESASOMBatchSubOrder mesASOMBatchSubOrder = new MESASOMBatchSubOrder();

mesASOMBatchSubOrder.setOrderno(OrderNo);

mesASOMBatchSubOrder.setPlantno(plantNo);

//TODO

mesASOMBatchSubOrder.setLineno("");

mesASOMBatchSubOrder.setPartno(partNo);

mesASOMBatchSubOrder.setPlanproductdate(PCContext.getFunctions().getDBTime());

mesASOMBatchSubOrder.setQty(count);

mesASOMBatchSubOrder.setSublinetype(subLineType);

mesASOMBatchSubOrder.setJobno(Long.valueOf(jobNo));

mesASOMBatchSubOrder.setSequence(sequence);

mesASOMBatchSubOrder.setSource(OrderSourceStatus.PROD);

mesASOMBatchSubOrder.setStatus(0L);

Response response = mesASOMBatchSubOrder.save(null, null, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMBatchSubOrder success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMBatchSubOrder fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMBatchSubOrder", keyWord });

}

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

public String getSubSequenceBySubLineType(String subLineType) throws MESException {

String functionName = "getSubSequenceBySubLineType(String subLineType)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

//通过分线类型获取车间

Area shop = getShopBySubLineType(subLineType);

String keyWord = "subLineType:[" +subLineType + "],shop:[" +shop + "]";

//通过车间和分线类型获取UserSequence

Response nextValueResponse = getSubOrderSequenceResponse(shop,subLineType);

if(nextValueResponse.isError())

{

LogUtility.logError(MODEL\_NAME, "get UserSequence faild,Error Info["+nextValueResponse.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

UserSequenceValue usValue = (UserSequenceValue) nextValueResponse.getResult();

int seq = usValue.getValue();

String sequence = "S" + String.format("%8d", seq).replace(" ", "0");

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return sequence;

}

public String getOrderNoBySubLineType(String subLineType) throws MESException {

String functionName = "getOrderNoBySubLineType(String subLineType)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

String year = TimeHelper.timeToString(PCContext.getFunctions().getDBTime(), "YYYY");

String keyWord = "subLineType:[" +subLineType + "],year:[" +year + "]";

Response nextValueResponse = getSubOrderNoSequenceResponse(subLineType,year);

if(nextValueResponse.isError())

{

LogUtility.logError(MODEL\_NAME, "get UserSequence faild,Error Info["+nextValueResponse.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

throw new MESException("save order faild, keyWord<" + keyWord + ">");

}

UserSequenceValue usValue = (UserSequenceValue) nextValueResponse.getResult();

int sequence = usValue.getValue();

String number = String.format("%6d", sequence).replace(" ", "0");

String orderNo = subLineType + year + number;

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return orderNo;

}

public Area getShopBySubLineType(String subLineType) {

// TODO Auto-generated method stub

Area area = PCContext.getFunctions().getAreaByName("W6");

return area;

}

public Response getSubOrderSequenceResponse(Area shop,String subLineType)

{

String sequenceName = "SUB\_ORDER\_SEQUENCE\_" + shop.getName() + "\_" + subLineType;

UserSequence orderShopSequence = PCContext.getFunctions().getUserSequenceByName(sequenceName);

if (orderShopSequence == null) {

orderShopSequence = PCContext.getFunctions().createUserSequence(sequenceName);

orderShopSequence.setInitialValue(0);

orderShopSequence.setMaxValue(Integer.MAX\_VALUE);

orderShopSequence.setIncrementValue(1);

orderShopSequence.save();

}

Response nextValueResponse = orderShopSequence.getNextValue();

return nextValueResponse;

}

@Override

public Response getSubOrderNoSequenceResponse(String subLineType, String year)

{

String sequenceName = subLineType + year;

UserSequence subOrderNoSequence = PCContext.getFunctions().getUserSequenceByName(sequenceName);

if (subOrderNoSequence == null) {

subOrderNoSequence = PCContext.getFunctions().createUserSequence(sequenceName);

subOrderNoSequence.setInitialValue(0);

subOrderNoSequence.setMaxValue(Integer.MAX\_VALUE);

subOrderNoSequence.setIncrementValue(1);

subOrderNoSequence.save();

}

Response nextValueResponse = subOrderNoSequence.getNextValue();

return nextValueResponse;

}

@Override

public void updateMESASOMBIWSubOrder(Long atrKey) throws DatasweepException, MESException

{

String functionName = "updateMESASOMBIWSubOrder(Long atrKey)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

String keyWord = "atrKey:[" +atrKey + "]";

IMESASOMBIWSubOrder mesASOMBIWSubOrder = new MESASOMBIWSubOrder(atrKey);

mesASOMBIWSubOrder.setIsbroadcasted(true);

Response response = mesASOMBIWSubOrder.save(null, String.valueOf(atrKey), null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMBIWSubOrder success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMBIWSubOrder fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMBIWSubOrder", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

@Override

public Response updateMESASOMBIWSubOrderPnuts(Long atrKey)

{

Response resp = new Response();

try

{

updateMESASOMBIWSubOrder(atrKey);

}

catch (Exception e)

{

resp.addError(

new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public Response generateSyncSubOrderByBIWSubOrderPnuts()

{

Response resp = new Response();

try

{

generateSyncSubOrderByBIWSubOrder();

} catch (Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

private void generateSyncSubOrderByBIWSubOrder()

{

String functionName = "generateSyncSubOrderByBIWSubOrder()";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

MESASOMBIWSubOrderFilter subOrderFilter = new MESASOMBIWSubOrderFilter();

try

{

subOrderFilter.forIsbroadcastedEqualTo(false);

subOrderFilter.addOrderBy(MESASOMBIWSubOrder.COL\_NAME\_SEQUENCE, IATRowFilterAttributes.ATCOLUMN,

IFilterSortOrders.ASCENDING);

List<IMESASOMBIWSubOrder> subOrders = subOrderFilter.getFilteredObjects();

if (null != subOrders && subOrders.size() > 0)

{

for (IMESASOMBIWSubOrder subOrder : subOrders)

{

String sub\_line\_type = subOrder.getZone();

boolean isSyncOrder = isSyncOrderType(sub\_line\_type);

if(isSyncOrder)

{

MESASOMSyncSubOrder syncSubOrder = new MESASOMSyncSubOrder();

Order order = subOrder.getShoporder().getOrder();

Area shop = subOrder.getShop();

String keyWord = "keyWord: orderNo[" + order.getOrderNumber() + "], Zone[" + sub\_line\_type +"]";

//生成顺序号

Long sequence = getSequenceForPropert(order);

if(null == sequence)

{

LogUtility.logError(MODEL\_NAME, "get sequence faild, keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

continue ;

}

//获取job\_no

IMESASSMSubLineType subLineType = getSubLineTypeByName(sub\_line\_type);

Response StrJobNo = generateSubLineJobNoPnuts(order,subLineType);

String jobNo = (String) StrJobNo.getResult();

if (null == jobNo || jobNo.equals(""))

{

LogUtility.logError(MODEL\_NAME, "get jobNo faild,Error Info["+StrJobNo.getFirstErrorMessage()+"], keyWord<" + keyWord + ">", PCContext.getFunctions().createTime(), functionName, functionName);

continue ;

}

syncSubOrder.setSublinetype(subOrder.getZone());

syncSubOrder.setOrderno(order.getOrderNumber());

syncSubOrder.setVin(OrderHelperEx.getVinFromProperty(order));

syncSubOrder.setRfid(OrderHelperEx.getRFID(order));

syncSubOrder.setPlantno(shop.getName());

syncSubOrder.setLineno(""); //暂时不用生产线

syncSubOrder.setPartno((String) order.getUDA("part\_number")); // 暂用整车物料号

syncSubOrder.setPlanproductdate(subOrder.getPlanstartdate());

syncSubOrder.setPlanofflinedate(subOrder.getPlancompletiondate());

syncSubOrder.setJobno(Long.valueOf(jobNo));

syncSubOrder.setSequence(String.valueOf(sequence));

syncSubOrder.setSource("PROD");

syncSubOrder.setStatus(0L);

syncSubOrder.setBroadcastetime(PCContext.getFunctions().createTime());

syncSubOrder.setIsbroadcasted(false);

syncSubOrder.Save(null, keyWord, null);

subOrder.setIsbroadcasted(true);

subOrder.setBroadcastetime(PCContext.getFunctions().createTime());

subOrder.Save(null, keyWord, null);

}

}

}

}

catch (DatasweepException e)

{

LogUtility.logError(MODEL\_NAME, e.getMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), "generate SyncSubOrder By BIWSubOrder failed!");

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

private Long getSequenceForPropert(Order order)

{

MESASOMOrderPropertyFilter propertyFilter = new MESASOMOrderPropertyFilter();

try

{

propertyFilter.forOrderEqualTo(order);

if (propertyFilter.getCount() > 0)

{

return propertyFilter.getFilteredObjects().get(0).getShopsequence();

}

}

catch (DatasweepException e)

{

e.printStackTrace();

}

return null;

}

private IMESASSMSubLineType getSubLineTypeByName(String sub\_line\_type)

{

IMESASSMSubLineTypeFilter subLineTypeFilter = new MESASSMSubLineTypeFilter();

try

{

subLineTypeFilter.forNameEqualTo(sub\_line\_type);

}

catch (DatasweepException e)

{

// TODO Auto-generated catch block

e.printStackTrace();

}

List<IMESASSMSubLineType> subLineTypes = subLineTypeFilter.getFilteredObjects();

if (null != subLineTypes && subLineTypes.size() > 0)

{

return subLineTypes.get(0);

}

return null;

}

private boolean isSyncOrderType(String sub\_line\_type)

{

MESASSMSubLineTypeFilter subLineTypeFilter = new MESASSMSubLineTypeFilter();

try

{

subLineTypeFilter.forNameEqualTo(sub\_line\_type);

List<IMESASSMSubLineType> subLines = subLineTypeFilter.getFilteredObjects();

if (null != subLines && subLines.size() >0)

{

MESASSMSubLineType subLine = (MESASSMSubLineType) subLines.get(0);

Long type = subLine.getType();

if (type == 20L)

{

return true;

}

}

}

catch (DatasweepException e)

{

e.printStackTrace();

}

return false;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* zzw 2020/6/6

\* generate sub line jobno

\* when biwsuborder insert into AS\_OM\_SyncSubOrder

\* @param workOrder

\* @param productionLine

\* @return

\* @throws DatasweepException

\* @throws MESException

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

@Override

public String generateSubLineJobNo(Order workOrder, IMESASSMSubLineType subLineType)

throws DatasweepException, MESException

{

String functionName = "generateSubLineJobNo(Order workOrder, IMESASSMSubLineType subLineType)";

LogUtility.logInfo("OM", "Start Generate Sub Order Job No Function", PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if(workOrder == null)

{

LogUtility.logError("OM", "Given workOrder is null!", PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("MES\_MSG\_ERROR", "GIVEN\_PARAMETER\_IS\_NULL\_ERROR", new Object[] {

"Order"

});

}

if(subLineType == null)

{

LogUtility.logError("OM", "Given subLineType is null!", PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("MES\_MSG\_ERROR", "GIVEN\_PARAMETER\_IS\_NULL\_ERROR", new Object[] {

"subLineType"

});

}

String keyWord = (new StringBuilder("Order Number:[")).append(workOrder.getOrderNumber()).append("], subLineType name[").append(subLineType.getName()).append("];").toString();

LogUtility.logInfo("OM", (new StringBuilder("Function Parameter values,KeyWord<")).append(keyWord).append(">").toString(), PCContext.getFunctions().createTime(), getClass().getName(), functionName);

String platformStr = (String)workOrder.getUDA("model");

IMESASSMPlatform platform = getSystemService().getPlatformObj(platformStr);

if(null == platform)

{

LogUtility.logError("OM", "Get platform is null!", PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("Get platform is null when generate sub line jobno");

}

MESASOMJobNoSubLineFilter jnsfilter = new MESASOMJobNoSubLineFilter();

jnsfilter.forPlatformEqualTo(platform);

jnsfilter.forSublinetypeEqualTo(subLineType);

List<IMESASOMJobNoSubLine> list = jnsfilter.getFilteredObjects();

IMESASOMFeatureGroup fg = null;

String jobNoLowStr = "";

if(list.size()>0)

{

fg = list.get(0).getFeaturegroup();

}

else

{

LogUtility.logError("OM", "Get featuregroup is null!", PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException("Get featuregroup is null when generate sub line jobno");

}

jobNoLowStr = getSystemService().getFeatureGroupValue(workOrder, "JOBNO", fg.getName());

//String jobNoHighStr = getSystemService().getFeatureGroupValue(workOrder, "JOBNO\_HIGH", jobNoObj.getFeaturegroup().getName());

LogUtility.logInfo("OM", "Get Sub Line Job No Function", PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return jobNoLowStr;

}

@Override

public Response generateSubLineJobNoPnuts(Order workOrder, IMESASSMSubLineType subLineType)

{

Response resp = new Response();

try

{

String jobNo = generateSubLineJobNo(workOrder, subLineType);

resp.setResult(jobNo);

}

catch(Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public IMESASOMElectroOrder getElectroOrderByVin(String vin) throws DatasweepException

{

IMESASOMElectroOrder electroOrder = null;

IMESASOMElectroOrderFilter electroOrderFilter = new MESASOMElectroOrderFilter();

electroOrderFilter.forVinEqualTo(vin);

List<IMESASOMElectroOrder> electroOrderList = electroOrderFilter.getFilteredObjects();

if(electroOrderList.size()>0)

{

electroOrder = electroOrderList.get(0);

}

return electroOrder;

}

@Override

public Response getElectroOrderByVinPnuts(String vin)

{

Response resp = new Response();

try

{

IMESASOMElectroOrder electroOrder = getElectroOrderByVin(vin);

resp.setResult(electroOrder);

}

catch(Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public IMESASOMBDCStock getBDCStockByVin(String area,String vin) throws DatasweepException

{

IMESASOMBDCStock bdcStock = null;

IMESASOMBDCStockFilter bdcStockFilter = new MESASOMBDCStockFilter();

bdcStockFilter.forAreaEqualTo(area);

bdcStockFilter.forVinEqualTo(vin);

List<IMESASOMBDCStock> bdcStockList = bdcStockFilter.getFilteredObjects();

if(bdcStockList.size()>0)

{

bdcStock = bdcStockList.get(0);

}

return bdcStock;

}

@Override

public Response getBDCStockByVinPnuts(String area,String vin)

{

Response resp = new Response();

try

{

IMESASOMBDCStock bdcStock = getBDCStockByVin(area,vin);

resp.setResult(bdcStock);

}

catch(Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public IMESASOMBDCOutQueue getBDCOutQueue(String area,String vin) throws DatasweepException

{

IMESASOMBDCOutQueue bdcOutQueueObj = null;

IMESASOMBDCOutQueueFilter bdcOutQueueFilter = new MESASOMBDCOutQueueFilter();

bdcOutQueueFilter.forAreaEqualTo(area);

bdcOutQueueFilter.forVinEqualTo(vin);

List<IMESASOMBDCOutQueue> bdcOutQueueList = bdcOutQueueFilter.getFilteredObjects();

if(bdcOutQueueList.size()>0)

{

bdcOutQueueObj = bdcOutQueueList.get(0);

}

return bdcOutQueueObj;

}

@Override

public Response getBDCOutQueuePnuts(String area,String vin)

{

Response resp = new Response();

try

{

IMESASOMBDCOutQueue getBDCOutQueue = getBDCOutQueue(area,vin);

resp.setResult(getBDCOutQueue);

}

catch(Exception e)

{

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

public void calcBDCOutQueueForWBS() throws DatasweepException, MESException

{

String functionName = "calcBDCOutQueueForWBS()";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

String area = BDCAreaName.WBS;

//1.获取已计算到出车队列，但未出车的车辆数

IMESASOMBDCOutQueueFilter bdcOutQueueFilter = new MESASOMBDCOutQueueFilter();

bdcOutQueueFilter.forAreaEqualTo(area);

bdcOutQueueFilter.forIsoutEqualTo(false);

Long notOutCount = bdcOutQueueFilter.getCount();

//TODO 2.获取出车队列开始计算数量

Long calcOutOueueCount = 100l;

if(notOutCount <= calcOutOueueCount)

{

Long vehicleCalcNum = 0l;

Long electroOrderCalcNum = 0l;

//3.获取保留车列表

IMESASOMBDCHoldVehicleFilter holdVehicleFilter = new MESASOMBDCHoldVehicleFilter();

holdVehicleFilter.forIsholdEqualTo(true);

List<IMESASOMBDCHoldVehicle> holdVehicleList = holdVehicleFilter.getFilteredObjects();

//TODO 4.获取每次计算数量

Long calcNum = getBDCOutQueueCalcNumber();

//5.获取未排到出车队列电泳件订单数量和电泳件订单列表

MESASOMBDCStockFilter bdcStockFilter = new MESASOMBDCStockFilter();

bdcStockFilter.forIsinqueueEqualTo(false);

bdcStockFilter.forAreaEqualTo(area);

bdcStockFilter.forIselectroorderEqualTo(true);

bdcStockFilter.addOrderBy(IATRowFilterAttributes.CREATIONTIME, IFilterSortOrders.ASCENDING);

Long electroOrderCount = bdcStockFilter.getCount();

List<IMESASOMBDCStock> electroOrderStockList = bdcStockFilter.getFilteredObjects();

//5.获取未排到出车队列的车辆列表（包含冻结车辆）

bdcStockFilter = new MESASOMBDCStockFilter();

bdcStockFilter.forIsinqueueEqualTo(false);

bdcStockFilter.forAreaEqualTo(area);

bdcStockFilter.forIselectroorderEqualTo(false);

bdcStockFilter.addOrderBy(MESGeneratedASOMBDCStock.COL\_NAME\_BSN, IATRowFilterAttributes.ATCOLUMN, IFilterSortOrders.ASCENDING);

List<IMESASOMBDCStock> bdcStockList = bdcStockFilter.getFilteredObjects();

//6.从车辆列表中去除冻结车辆

for(int j=0;j<holdVehicleList.size();j++)

{

IMESASOMBDCHoldVehicle holdVehicle = holdVehicleList.get(j);

Iterator<IMESASOMBDCStock> bdcStock=bdcStockList.iterator();

while(bdcStock.hasNext()){

IMESASOMBDCStock Stock=bdcStock.next();

if (holdVehicle.getOrder().getKey() == Stock.getOrder().getKey()) {

bdcStock.remove();

}

}

}

//7.获取未排到出车队列的车辆列表（不包冻结车辆）

Long bdcStockCount = bdcStockFilter.getCount();

//8.获取计算的数量

if(calcNum>=(electroOrderCount+bdcStockCount))

{

vehicleCalcNum = bdcStockCount;

electroOrderCalcNum = electroOrderCount;

}

else if(electroOrderCount>0)

{

if(bdcStockCount>=(calcNum-1))

{

vehicleCalcNum = calcNum-1;

electroOrderCalcNum = 1l;

}

else

{

vehicleCalcNum = bdcStockCount;

electroOrderCalcNum = calcNum - bdcStockCount;

}

}

else

{

vehicleCalcNum = calcNum;

if(vehicleCalcNum>bdcStockCount)

{

vehicleCalcNum = bdcStockCount;

}

}

//=============================================================================

List<String> profileNameList = new ArrayList<String>();

//获取BDC\_Ration信息<名词key，对象>

Map<Long, IMESASOMBDCAttrGroup> bdcAttrGroupMap = new HashMap<Long,IMESASOMBDCAttrGroup>();

Map<Long, Map<String,String>> orderProfileMap = new HashMap<Long, Map<String,String>>();

Map<Long, Long> maxRatioQtyMap = new HashMap<Long, Long>();

Map<Long, Long> currRatioQtyMap = new HashMap<Long, Long>();

Map<Long, Map<Long,Boolean>> orderRatioMap = new HashMap<Long, Map<Long,Boolean>>();

IMESASOMBDCRatioFilter bdcRatioFilter = new MESASOMBDCRatioFilter();

bdcRatioFilter.forAreaEqualTo(area);

bdcRatioFilter.forIsactiveEqualTo(true);

List<IMESASOMBDCRatio> bdcRatioList = bdcRatioFilter.getFilteredObjects();

for(IMESASOMBDCRatio bdcRatio:bdcRatioList)

{

Long maxQty = 1l;

bdcAttrGroupMap.put(bdcRatio.getAttrgroupObj().getKey(),bdcRatio.getAttrgroupObj());

Long ratioQty = bdcRatio.getRatioqty();

Long totalQty = bdcRatio.getTotalqty();

if(totalQty>0)

{

maxQty = (vehicleCalcNum + electroOrderCalcNum)\*ratioQty/totalQty;

}

if(maxQty<1)

{

maxQty = 1l;

}

maxRatioQtyMap.put(bdcRatio.getKey(), maxQty);

currRatioQtyMap.put(bdcRatio.getKey(), 0l);

}

for(IMESASOMBDCAttrGroup attrGroup : bdcAttrGroupMap.values())

{

List<MESASOMBDCAttrGroupItem> attrGroupItemList = attrGroup.getBDCAttrGroupItemList();

for(MESASOMBDCAttrGroupItem attrGroupItem:attrGroupItemList)

{

Boolean isFind = false;

//获取配置项字表名

String profile = attrGroupItem.getProfile();

for(String profileName:profileNameList)

{

if(profileName.equals(profile))

{

isFind = true;

break;

}

}

if(isFind == false)

{

profileNameList.add(profile);

}

}

}

List<IMESASOMBDCStock> outQueueTempList = new ArrayList<IMESASOMBDCStock>();

IMESASOMBDCStock prevBdcStock = null;

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

{

Boolean isAdd = false;

if(prevBdcStock == null)

{

prevBdcStock = bdcStockList.get(0);

bdcStockList.remove(prevBdcStock);

Map<Long,Boolean> ratioMapOfOrder = getOrderRatioMapForOrder(orderRatioMap,prevBdcStock.getOrder(), profileNameList,bdcRatioList);

addBDCStockToOutQueueList(prevBdcStock,outQueueTempList,ratioMapOfOrder,currRatioQtyMap);

continue;

}

IMESASOMBDCStock bdcStockColorOfFirst = null;

String prevPaintcolor = prevBdcStock.getPaintcolor();

Iterator<IMESASOMBDCStock> Stock =bdcStockList.iterator();

while(Stock.hasNext()){

IMESASOMBDCStock bdcStock=Stock.next();

String paintcolor = bdcStock.getPaintcolor();

if (prevPaintcolor.equals(paintcolor)) {

Map<Long,Boolean> ratioMapOfOrder = getOrderRatioMapForOrder(orderRatioMap, bdcStock.getOrder(), profileNameList,bdcRatioList);

Boolean isExceed = exceedRation(ratioMapOfOrder,maxRatioQtyMap,currRatioQtyMap);

if(isExceed==false)

{

prevBdcStock = bdcStock;

Stock.remove();

addBDCStockToOutQueueList(prevBdcStock,outQueueTempList,ratioMapOfOrder,currRatioQtyMap);

isAdd = true;

// break;

}

else

{

if(bdcStockColorOfFirst == null)

{

bdcStockColorOfFirst = bdcStock;

}

}

}

}

//找其它颜色，不超占比

for(IMESASOMBDCStock bdcStock:bdcStockList)

{

String paintcolor = bdcStock.getPaintcolor();

if(! prevPaintcolor.equals(paintcolor))

{

Map<Long,Boolean> ratioMapOfOrder = getOrderRatioMapForOrder(orderRatioMap, bdcStock.getOrder(), profileNameList,bdcRatioList);

Boolean isExceed = exceedRation(ratioMapOfOrder,maxRatioQtyMap,currRatioQtyMap);

if(isExceed==false)

{

prevBdcStock = bdcStock;

bdcStockList.remove(prevBdcStock);

addBDCStockToOutQueueList(prevBdcStock,outQueueTempList,ratioMapOfOrder,currRatioQtyMap);

isAdd = true;

break;

}

}

}

//找同一颜色超占比

if(isAdd==false)

{

if(bdcStockColorOfFirst != null )

{

prevBdcStock = bdcStockColorOfFirst;

bdcStockList.remove(prevBdcStock);

Map<Long,Boolean> ratioMapOfOrder = getOrderRatioMapForOrder(orderRatioMap,prevBdcStock.getOrder(), profileNameList,bdcRatioList);

addBDCStockToOutQueueList(prevBdcStock,outQueueTempList,ratioMapOfOrder,currRatioQtyMap);

continue;

}

}

else

{

continue;

}

if(bdcStockList.size()>0)

{

prevBdcStock = bdcStockList.get(0);

bdcStockList.remove(prevBdcStock);

Map<Long,Boolean> ratioMapOfOrder = getOrderRatioMapForOrder(orderRatioMap,prevBdcStock.getOrder(), profileNameList,bdcRatioList);

addBDCStockToOutQueueList(prevBdcStock,outQueueTempList,ratioMapOfOrder,currRatioQtyMap);

}

}

// for(int i=0;i<electroOrderCalcNum;i++)

// {

// outQueueTempList.add(electroOrderStockList.get(i));

// }

Long groupNo = 0l;

if(outQueueTempList.size()>0)

{

//生成分组号

groupNo = SystemHelperEx.getUserSequenceResponseByName(area + "\_OUT\_QUEUE");

Long mesSeq=0l;

for(IMESASOMBDCStock bdcStockObj:outQueueTempList)

{

if(mesSeq<calcNum-electroOrderCalcNum)

{

mesSeq = mesSeq + 1l;

addOrderTobdcOutQueue( bdcStockObj, groupNo, mesSeq);

if(electroOrderCalcNum>0)

{

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

{

bdcStockObj = electroOrderStockList.get(i);

mesSeq = mesSeq + 1l;

addOrderTobdcOutQueue( bdcStockObj, groupNo, mesSeq);

}

}

}

}

}

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

private Map<Long,Boolean> getOrderRatioMapForOrder(Map<Long, Map<Long,Boolean>> orderRatioMap,Order order,List<String> profileNameList,List<IMESASOMBDCRatio> bdcRatioList) throws MESException, DatasweepException

{

Long orderKey = order.getOrderKey();

Map<Long,Boolean> ratioMapOfOrder = orderRatioMap.get(orderKey);

if(ratioMapOfOrder == null)

{

ratioMapOfOrder = new HashMap<Long,Boolean>();

Map<String,String> profileDataMap = getSystemServiceEx().getProfileDataMap(order, "BDC", profileNameList);

for(IMESASOMBDCRatio bdcRatio:bdcRatioList)

{

IMESASOMBDCAttrGroup attrgroupObj = bdcRatio.getAttrgroupObj();

ratioMapOfOrder.put(bdcRatio.getKey(), isAccordAttrOfOrder(attrgroupObj, profileDataMap));

}

orderRatioMap.put(orderKey, ratioMapOfOrder);

}

return ratioMapOfOrder;

}

private Boolean exceedRation(Map<Long,Boolean> ratioMapOfOrder,Map<Long, Long> maxRatioQtyMap,Map<Long, Long> currRatioQtyMap)

{

Boolean isExceed = false;

Set<Long> keys = ratioMapOfOrder.keySet();

//判断长工时

for (Long key : keys) {

Boolean isAccord = ratioMapOfOrder.get(key);

if(isAccord == true)

{

//此处逻辑有问题 计算下来少一个 应该是大于 而不能是等于

Long maxRatioQty = maxRatioQtyMap.get(key);

Long currRatioQty = currRatioQtyMap.get(key);

if(currRatioQty >= maxRatioQty)

{

isExceed = true;

break;

}

}

}

return isExceed;

}

public void addBDCStockToOutQueueList(IMESASOMBDCStock bdcStock,List<IMESASOMBDCStock> outQueueTempList,Map<Long,Boolean> ratioMapOfOrder,Map<Long, Long> currRatioQtyMap)

{

Set<Long> keys = ratioMapOfOrder.keySet();

for (Long key : keys) {

Boolean isAccord = ratioMapOfOrder.get(key);

if(isAccord == true)

{

Long currRatioQty = currRatioQtyMap.get(key);

currRatioQtyMap.put(key, currRatioQty + 1l);

}

}

outQueueTempList.add(bdcStock);

}

public IMESASOMBDCOutQueue addOrderTobdcOutQueue(IMESASOMBDCStock bdcStock, Long groupNo, Long mesSeq) throws MESException

{

String functionName = "addOrderTobdcOutQueue(IMESASOMBDCStock bdcStock, Boolean isManual)";

LogUtility.logInfo(

MODEL\_NAME, "Start " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

if (bdcStock == null)

{

LogUtility.logError(

MODEL\_NAME, "GIVEN bdcStock is null!",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR,

IMessage.MESSAGE\_ID\_GIVEN\_PARAMETER\_NOT\_NULL, new Object[] { "bdcStock" });

}

Order order = bdcStock.getOrder();

String keyWord = "area["+bdcStock.getArea()+"],OrderNumber:[" + bdcStock.getOrdernumber() + "]";

bdcStock.setIsinqueue(true);

Response response = bdcStock.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMBDCStock success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

}

else

{

LogUtility.logError(MODEL\_NAME,

"Save MESASOMBDCStock fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMBDCStock", keyWord });

}

Long sortNo = SystemHelperEx.getUserSequenceResponseByName(bdcStock.getArea()+"\_OUT\_QUEUE");

IMESASOMBDCOutQueue bdcOutQueue = new MESASOMBDCOutQueue();

bdcOutQueue.setArea(bdcStock.getArea());

bdcOutQueue.setBdcseq(0l);

bdcOutQueue.setBsn(bdcStock.getBsn());

bdcOutQueue.setGroupno(groupNo);

bdcOutQueue.setIselectroorder(bdcStock.getIselectroorder());

bdcOutQueue.setIsmanual(false);

bdcOutQueue.setIsout(false);

bdcOutQueue.setMesseq(mesSeq);

bdcOutQueue.setOrder(bdcStock.getOrder());

bdcOutQueue.setOrdernumber(bdcStock.getOrdernumber());

bdcOutQueue.setPaintcolor(bdcStock.getPaintcolor());

bdcOutQueue.setRequestcount(0l);

bdcOutQueue.setSendstatus(0l);

bdcOutQueue.setSortno(sortNo);

bdcOutQueue.setVin(bdcStock.getVin());

response = bdcOutQueue.save(null, keyWord, null);

if (response.isOk())

{

LogUtility.logInfo(MODEL\_NAME, "Save MESASOMBDCOutQueue success,KeyWord<" + keyWord + ">",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

} else

{

bdcStock.setIsinqueue(false);

response = bdcStock.save(null, keyWord, null);

LogUtility.logError(MODEL\_NAME,

"Save MESASOMBDCOutQueue fail,KeyWord<" + keyWord + ">,ERROR INFO:"

+ response.getFirstErrorMessage(),

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

throw new MESException(IMessage.MESSAGE\_PACK\_NAME\_ERROR, IMessage.MESSAGE\_ID\_SAVE\_OBJECT\_FAILURE,

new Object[]

{ "MESASOMBDCOutQueue", keyWord });

}

LogUtility.logInfo(MODEL\_NAME, "Finish " + functionName + " function...",

PCContext.getFunctions().createTime(), getClass().getName(), functionName);

return bdcOutQueue;

}

//获取拉入拉出标志表状态

@Override

public IMESASOMTakeInOutStatus getTakeInOutStatusObj(String bsn) throws DatasweepException

{

IMESASOMTakeInOutStatus takeInOutStatusObj = null;

IMESASOMTakeInOutStatusFilter takeInOutStatusFilter = new MESASOMTakeInOutStatusFilter();

takeInOutStatusFilter.forBsnEqualTo(bsn);

List<IMESASOMTakeInOutStatus> takeInOutStatusList = takeInOutStatusFilter.getFilteredObjects();

if(takeInOutStatusList.size()>0)

{

takeInOutStatusObj = takeInOutStatusList.get(0);

}

return takeInOutStatusObj;

}

@Override

public Response getTakeInOutStatusObjPnuts(String bsn) {

Response resp = new Response();

try {

resp.setResult(getTakeInOutStatusObj( bsn));

} catch (Exception e) {

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

//

@Override

public Long getTakeInOutStatusForVehicle(String bsn) throws DatasweepException

{

Long status = TakeInOutStatus.IN;

IMESASOMTakeInOutStatus takeInOutStatusObj = getTakeInOutStatusObj(bsn);

if(takeInOutStatusObj!=null)

{

status = takeInOutStatusObj.getStatus();

}

return status;

}

@Override

public Response getTakeInOutStatusForVehiclePnuts(String bsn) {

Response resp = new Response();

try {

resp.setResult(getTakeInOutStatusForVehicle( bsn));

} catch (Exception e) {

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

//更新take in out标志表

@Override

public Boolean updateTakeInOutStatus(String bsn,Long status) throws DatasweepException

{

Boolean updateResult = true;

IMESASOMTakeInOutStatus takeInOutStatusObj = getTakeInOutStatusObj(bsn);

if(takeInOutStatusObj==null)

{

takeInOutStatusObj = new MESASOMTakeInOutStatus();

takeInOutStatusObj.setBsn(bsn);

}

Response response = takeInOutStatusObj.save(null, null, null);

if(response.isError())

{

updateResult = false;

}

return updateResult;

}

@Override

public Response updateTakeInOutStatusPnuts(String bsn,Long status) {

Response resp = new Response();

try {

resp.setResult(updateTakeInOutStatus( bsn, status));

} catch (Exception e) {

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public Boolean isPlanTakeOut(String bsn,Station station) throws DatasweepException

{

Boolean result = false;

IMESASOMOrderRoutePlanFilter orderRoutePlanFilter = new MESASOMOrderRoutePlanFilter();

orderRoutePlanFilter.forBsnEqualTo(bsn);

orderRoutePlanFilter.forRoutestationEqualTo(station);

orderRoutePlanFilter.forIsactiveEqualTo(true);

if(orderRoutePlanFilter.getCount()>0)

{

result = true;

}

return result;

}

@Override

public Response isPlanTakeOutPnuts(String bsn,Station station) {

Response resp = new Response();

try {

resp.setResult(isPlanTakeOut( bsn, station));

} catch (Exception e) {

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public Boolean isExistTakeOut(String bsn) throws DatasweepException

{

Boolean isExist = false;

IMESASOMTakeInOutFilter filter = new MESASOMTakeInOutFilter();

filter.forBsnEqualTo(bsn);

filter.forStatusEqualTo(TakeInOutStatus.OUT);

if(filter.getCount()>0)

{

isExist = true;

}

return isExist;

}

@Override

public Response isExistTakeOutPnuts(String bsn) {

Response resp = new Response();

try {

resp.setResult(isExistTakeOut( bsn));

} catch (Exception e) {

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

@Override

public Boolean isStationType(Station station,String stationType) throws DatasweepException

{

Boolean result = false;

IMESASSMStationTypeFilter filter = new MESASSMStationTypeFilter();

filter.forStationEqualTo(station);

filter.forTypeEqualTo(stationType);

if(filter.getCount()>0)

{

result = true;

}

return result;

}

@Override

public Response isStationTypePnuts(Station station,String stationType) {

Response resp = new Response();

try {

resp.setResult(isStationType( station,stationType));

} catch (Exception e) {

resp.addError(new Error(e, PCContext.getServerImpl()));

}

return resp;

}

}