Hello All, Good Afternoon, This is Vinayak Indulkar, I’m part of PiP team, or you can say CAP-iPOS team. I have Nandhakumar and Shilpa alongside with me. We 3 will explain today about PiP application.

Before proceeding with iPOS high level overview and PiP components. Let us know what is PiP exactly?

PiP stands for Point Of Sale Integration Point, it is an integration application residing on the TP.Net server which facilitate the communication between POS and IKEA legacy systems such as MHS, iSELL, SAMS etc.

* If you see this iPOS integration diagram, PiP is communicating with many systems which are responsible for central operations, store operations, article management information, business intelligence, foods area.
* If you see this slide, many systems that are direct or indirectly connected to PiP is shown in this high level architecture diagram. here iKEA legacy systems are classified as store level and country/central level. Mention the systems.
* PiP is basically responsible for many jobs like customer card authorization, tender information, transforming the data, article import, article sales transactions, payments processing, returns transactions, end of day and generating store level and country specific reports. So in order to perform these operations, PiP have separate components which we can say as modules.
* These are the 11 components of PiP
* Item import – this component deals with item/article management. Item/article is nothing a product which we purchase it from iKEA stores. It can be a furniture’s, kitchen assembly, or any food item. So, the pos system needs to be loaded with item details such as item name, item price, item fee, item tax etc before it gets operational.

We have three source systems here, RIX, MHS, IIP. RIX stands for Retail Information Exchange, MHS stands for merchandize hub system, IIP stands for IKEA integration platform. We get 50 % of article details like item name, no of packages, taxitemcategorycode, productarea no & name, contract, co-worker, bistro prices from RIX. We get 40 % of article details like sellingruleid, unitcode, localno, vatcode, salestop details from MHS and remaining 10% of data like regular sales price, family discount price we get it from IIP.

Source for these 3 systems is Country Range Application, Pricera, SPA provides price, tax, fees for products to be sold at country level to store level. It is used to set up and administer Item range and price for retail units. So, CRA will push the country range info to RIX and store level info to IIP. Mode of transfer here is, IIP will push the details to PiP via Java Messaging service queue, queue name is pip.ebcpos.retailitemload.

MHS will use a EBC, enterprise business component, and service associated with ebc is called as business service, MHS will do item import to PiP via JMS queue that is pip.ebcpos.retailitemloadmhs.

Mode of transfer b/w rix and pip is done with help of adapter, ebcposrix which is facilitated by scheduled job which fetches the article info to jms queue pip.ebcposrix.retailitemload.

So once PiP gets the item details from these source systems, PiP then validates the data and set the source system flags in dB accordingly and qualify those items for pos transfer. PiP will merge the item details from 3 source systems & generate the item maintenance file to tp.net backstore office via jms topic pip.itemmaintenance. backstore office will subscribe to item maintenance topic to get latest item files from pip and then it’ll process them to all pos tills. The PIP application utilize the Log4J logging framework. PiP have real time logging mechanism. For this module, We have item import log and also we store latest item maintenance files which are sent successfully to backstore in lip samba location.

* Salesorder – is responsible for fetching of orders so that customer in store can pay for it. As we all know that orders are created in iSell application and for already created orders, the order details must be fetched in pos register to actually pay for it. So order fetching flow gets started when cashier scans a salesorder barcode in till, then tp.net will call the pip webservice getsalesorderbrief by providing the isell order number.

The pip salesorder application then map the request to mhs format and fetch the order content from mhs by calling their business service BSgetsalesorder. If order details is present in mhs, then pip will get successful response from MHS which contain order information, article information and that response will be sent to POS register. In case, if order is not present in mhs, then error response will be sent to pip. PiP will then send the request to iSell directly to retrieve the order information by calling their business service BSgetorderpayment via ITF webproxy call. So once pip receive the response from isell,The pip salesorder application will map the contents of order to tp.net format and send the response to pos register.

So, once the order fetching flow is successful, then customer can pay for an order by any payment methods like pay in advance, cash on delivery or pay through ikea credits.

Once the order is paid, POS completes the receipt transaction flow and sends the POSLog to PIP which I’ll be explaining in next slide.

PIP filter out salesorder transactions from POSLog and calls the iSell Business service Add Order Payment via ITF webproxy call to register the order payment in iSell. iSell has to acknowledge the request, if iSell accepts, the payment details will be saved in iSell. if iSell doesn’t acknowledge, then request will come back in PiP JMS topic pip.common.ReadyRequests. As per the existing functionality, Messages on PIP common topic is persisted to PIP database and managed manually by helpdesk using PIP Admin GUI. PiP has retry mechanism for failed orders where a scheduled job regularly resends failed transactions to the iSell after every 5 minutes until iSell accepts it or PiP clear the register payment request from JMS topic. I’d tell you why we have retry mechanism available at PiP. Since It’s a business critical case, When order has been paid, iSell application must know the order is paid & how much amount is paid.

Basically, Pip is acting as an intermediate system and forward the request and get the response and send it back to pos register. For this module, We do have salesorder and add payment logs for order fetching and register payment flow.

Receipt Transaction:

As the name implies, whenever cashier performs a transactions in ikea store, when the store is working, tp.net frontstore i.e pos register will be sending transaction files to tp.net backstore server. Pip will receive the sales transaction details from tp.net backstore server via jms topic pip.receipts.poslogtransaction

There are many systems that have have a dependency, to receive some sort of receipt data such as MHS, CLA, IIP, SPOSI.

Receipt subscribers also have different requirements on how often they like to receive data.

Two major types can be distinguished as

The line-by-line functionality provides and publishes all sales items purchased as single lines.

The receipt-by-receipt functionality provides and publishes all sales transactions as receipts.

So here PiP will consume the poslog data from tp.net backstore and filters out the requested data, transform it to acceptable format and sends them to these receipts subscribers.

MHS flow: PiP will transform the poslog arts format to ebcpayload format and calls the BSsavereceipt using EBCJMHSPOS01 to save the receipt in mhs dB. In case of void trx, cancelled trx, PiP will call Bsvoidreceipt to save the void receipt in mhs. log name is receipt transaction log where we can see which all trxs are sent to mhs during real time and to check if there is any failure. Failure may occur due to invalid data, bs call error or due to ebc problem which pip handles it via retry mechanism.

CLA / Common Landing Area flow:, will consume poslog and generate real time reports like SIMO(Sales Information Move Out). if there is any failure, downtime due to dB issue, PiP will handle it via timeout limit set in code and via retry mechanism. For this flow, we have clareceipt log

SPOSI flow: PiP filters out the food sales trx from poslog batch and process them by calling ws call to their VTS servers. Pip-food log to see if ws calls are completed successfully or not.

IIP flow: PiP will process the sales transaction details to IIP via JMS topics pip.SAPayment.validTransaction and pip.SAPayment.invalidTransaction. IIP will consume poslog data and generate many reports. IIP is currently managing information exchange to IKEA Retail Web and New Web Platform. For this flow , we have Receipttransaction valid, receipttransactioninvalid logs.

PiP application have logs retention policy of 30 days. And For china stores specifically we have logs retention policy of 90 days. PIP has implemented this change in such a way that Number of Days for storing logs is configured in PIP configuration file .We have Ansible scripts which is scheduled on the linux server to be executed on everyday basis to maintain the ikea file system space.