

## PkMS nTier MHE Development Overview.

Please note this document does not “replace” the PkMS documentation, rather it offers some additional “hints” and “suggestions”.

nTier provides communication to warehouse MHE equipment using two possible approaches:

1. Custom development of a PkMS Server process to communicate with the MHE directly using FTP, TCP/IP, or some other protocol and custom code hooks inside PkMS Servers.
2. Installation and configuration of the EIS product, also known as Comlink and custom code hooks inside PkMS Servers.

Using EIS is the preferred method, as the communications layer does not have to be developed. EIS provides TCP/IP, FTP, and XML file services. But in some highly custom environments, EIS may not be feasible.

### Important steps to take when defining a MHE interface

When defining a MHE interface several important steps must be taken to ensure success. These steps must be taken for each message and can be broken down as follows:

1. What protocol is the message going to be sent in? ( FTP, TCP/IP, XML )?
2. What is the direction of the message? Does it come from the MHE or from PkMS?
3. What is the exact event generating the message? Is it created while Packing a case? Is it generated by the MHE when releasing goods from an output spur?
4. What is the exact format of the message fields and length?
5. When the message is received, what action does the receiving process take?

After the interface and messages are defined, the work of customizing PkMS servers can begin. Typically, the approach used depends on whether or not EIS is going to be utilized as middle ware. The two sections below outline the two different paths:

### Design Approach without EIS

Design a custom server to “accept” messages from the MHE. This server accepts messages in the chosen format and validates them and then calls business logic to process the message. Messages may be processed using PkMS API's OR by directly calling a PkMS Server using CORBA. This server is multi-threaded, so one thread can receive and validate messages and other threads can process messages.

Design a custom client to “send” messages to the MHE. This “client” is also a PkMS Server. Typically, the client is given a CORBA interface that other PkMS Servers can call to generate a message when an event occurs. Then the message is placed on a queue to be sent to the MHE. This client is multi-threaded, in that the CORBA calls can generate multiple threads to place messages on the queue. A separate process is also required to send messages off the queue to the MHE.

### Design Approach with EIS

Design a custom server to “accept” messages from the EIS using the EIS API layer in PkMS. This layer allows the server to make a call and retrieve a message off the EIS queue. The server then validates the message and calls business logic to process the message. Messages may be processed using PkMS API's OR by directly calling a PkMS Server using CORBA. This server can be multi-threaded, so one thread can pull messages of the queue and other threads can process messages.

Design a custom client to “send” messages to the MHE. This “client” is also a PkMS Server. Typically, the client is given a CORBA interface that other PkMS Servers can call to generate a message when an event occurs. Then the message is placed on a queue to be sent to the EIS using the EIS API layer in PkMS. This client is multi-threaded, in that the CORBA calls can generate multiple threads to place messages on the queue. The EIS API layer is thread-safe, so no additional threads are required.

One additional step when using EIS is to Configure and Install EIS. For the initial design and development testing, EIS must be installed and all the messages, events, and end-points setup in EIS. Configuration of the messages in EIS should be documented for future configuration needs in other environments.

### Testing

EIS includes a program called “socktest” which can act as a TCP/IP server or client. To perform initial unit testing, or to test when a connection to the MHE is not available, socktest is a very valuable test tool. If EIS was not used, typically a small TCP/IP client/server program for testing can usually be developed from the skeleton of the custom server and client that were written to communicate with the MHE.