# Application Interworking(AIW)

Application Interworking (AIW) framework is a mechanism for dynamically adding features and functionality to applications without statically linking to the existing code, and without knowing in advance which services are available in the device.

A criteria item specifies one AIW service that a consumer application is interested in using. An array of criteria items is called interest. Consumer application's interest is defined in its resource file or it can be defined dynamically by using CAiwCriteriaItem. AIW framework uses criteria items to find out if there are suitable service providers available and then initializes the requested providers.

The main class of AIW framework is called CAiwServiceHandler. It implements consumer side of AIW API. Each application using AIW services must create an instance of that class.The Service Handler implements support for dynamically loadable service providers which offer services to consumer applications. The Service Handler maps consumers and service providers together via interests and data agreements and hides the consumers from the providers.

Example: If a third party application wants to make a Call, It can make use of the providers for that functionality. The consumer application specifies its interest in using such a service by providing the details in the CAiwCriteriaItem or in rss file as follows:

```
  // Specifying dynamically

  CAiwCriteriaItem* criteria = CAiwCriteriaItem::NewLC(
                                KAiwCmdCall, KAiwCmdCall,
                                _L8( "*" ) );


// using rss file

     RESOURCE AIW_INTEREST r_aiwconsumerapp_interest1
      {
           items =
           {
            AIW_CRITERIA_ITEM
             {
               id = EAIWPlaceHolder;
               serviceCmd = KAiwCmdCall;   //from aiwcommon.hrh
               serviceClass = KAiwClassMenu;   //from aiwcommon.hrh
               contentType = "*";
               //defaultProvider = 0x101F868E;
               maxProviders =1;
              }
            };
      }
```

This resource Id to the AttachL method of the service handler. AIW framework uses a generic parameter list for passing data between consumers and providers. Items on the generic parameter list have a semantic ID value, which defines the type of data the item holds. These sematic ID values are defined in the AIWGenericParams.hrh.

For Example when making a call, details like phonenumber,type of call(video,voice,voip),Name of the caller need to be passed. These values are passed using TAiwDialDataV1 and this is notified to the provider by specifying EGenericParamCallDialDataV1.

```
     CAiwServiceHandler* serviceHandler = CAiwServiceHandler::NewLC();

     // Create AIW interest
     RCriteriaArray interest;
     CleanupClosePushL( interest );

     CAiwCriteriaItem* criteria = CAiwCriteriaItem::NewLC(
                                  KAiwCmdCall, KAiwCmdCall,
                                  _L8( "*" ) );
     TUid base;
     base.iUid = KAiwClassBase;
     criteria->SetServiceClass( base );

     User::LeaveIfError( interest.Append( criteria ) );

     // Attach to AIW interest
     serviceHandler->AttachL( interest );

     // Create AIW param package
     TAiwDialDataV1 data;
     TAiwDialDataV1Pckg dataPckg( data );

     iNameBuffer = aName.Left( iNameBuffer.MaxLength() );

     data.SetName( iNameBuffer );

     iTelNumber = aNumber.Left( iTelNumber.MaxLength() );

      data.SetTelephoneNumber( iTelNumber );

     data.SetCallType( aCallType );

    data.SetWindowGroup( CCoeEnv::Static()->RootWin().Identifier() );

     CAiwGenericParamList& paramList = serviceHandler->InParamListL();

     TPtrC8 ptr;
     ptr.Set( dataPckg );
     TAiwVariant variant( ptr );

     TAiwGenericParam param( EGenericParamCallDialDataV1, variant );
```


(C) Copyright Nokia 2013. All rights reserved.

```
    paramList.AppendL( param );

    // Execute AIW command
    serviceHandler->ExecuteServiceCmdL( KAiwCmdCall, paramList,
                                serviceHandler->OutParamListL() );

    CleanupStack::PopAndDestroy( 3 ); // criteria, interest, serviceHandler
```


(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.