Component Definition Document (CDD) for the Publish-Subscribe Attachment Transfer Example Component Assembly

Rev. -

February 9, 2011

Prepared By:
Northrop Grumman Corporation
Electronic Systems
Baltimore, MD

Table of Contents

1 Introduction

1.1 Scope

This document captures the specification and design for the Publish-Subscribe Attachment Transfer Example software component assembly. This example assembly is targeted for deployment on the Scalable Node Architecture (SNA) real-time component framework. As such, it must be compliant with SNA Component Based Architecture (CBA) design guidelines.

This specification defines the component assembly's functional, interface and performance requirements, the context in which it must operate, and any design constraints it must adhere to. It provides criteria for verifying compliance, but it does not state methods for achieving results.

This is intended to be a relatively informal living document, to be included in same CM repository and package as the component source code. This CDD will initially be populated by a system engineer or software architect/lead to define component design constraints & guidelines. Over time, it will transition to enhance the "to be built" specification sections with "as built" design information documenting the final component product.

2 Applicable Documents

2.1 Applicable Government Documents

Document No.	Title

2.2 Other Applicable Documents

Document No.	Title

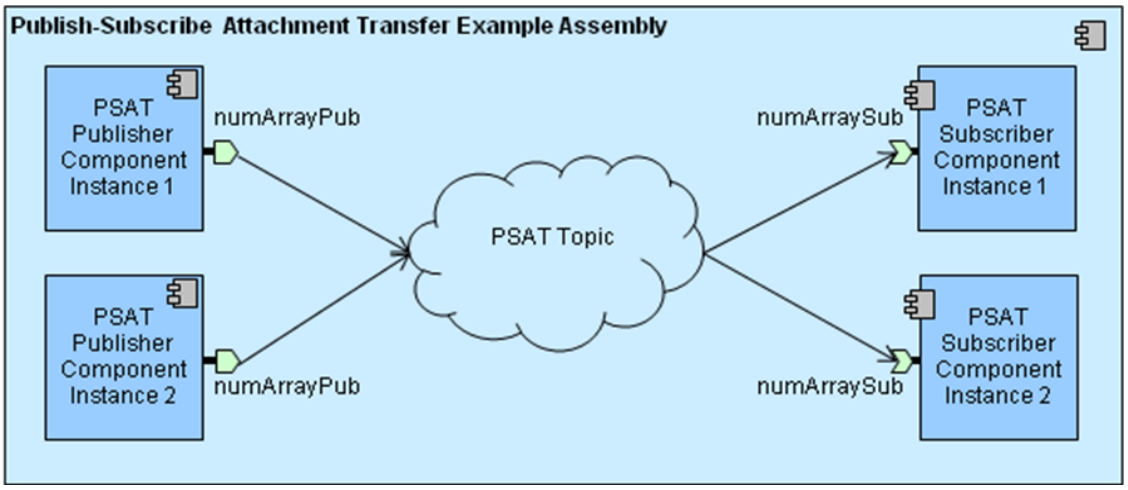
3 Component Description

3.1 Overview

The Publish-Subscribe Attachment Transfer Example component assembly is one of the component source examples included in the SNA SDK for reference, testing and experimentation. It illustrates a simple component-based publish-subscribe attachment transfer (PSAT) information exchange pattern utilizing two single-port monolithic components. Two instances of each component type (PSAT Publisher and Subscriber) are deployed. The PSAT Publisher component periodically publishes one or more attachments consisting of a large array of 64-bit numbers and the PSAT Subscriber component subscribes to and receives them. In addition, the PSAT Publisher component has an attribute which is used to calculate the numbers to place in the PSAT attachment. This is used to differentiate the output of each instance.

The assembly containing these two components is shown in Figure 3-1 below.

Figure 3-1 –Pub-Sub Attachment Transfer Example Component Assembly



This example assembly illustrates component executor composition showing code that uses the publish/subscribe attachment transfer (PSAT) pattern and associated event handlers.

3.2 Operational Context

The assembly is completely self-contained, with no external connections. It is designed to operate solely within the constraints of the SNA platform's development environment to allow a new user/developer to step through the SNA component based development (CBD) process of loading a component assembly into the SNA IDE, building/compiling it, and then executing it.

The example is provided with an appropriate set of SNA configuration files and a deployment plan to support its execution within a single-host SNA SDK "localhost" Virtual Machine (VM). Alternative variations on the default supplied design and deployment are possible via experimentation by a software developer.

4 Component Interfaces

The Publish-Subscribe Attachment Transfer Example assembly defines a single PSAT topic between a publisher port on each of the two instances of the PSAT Publisher monolithic component and a subscriber port on each of the PSAT Subscriber monolithic component, as shown in Figure 3-1. The component assembly has no external interfaces.

4.1 Service Ports

There are no internal or external client ports defined for this component assembly.

4.2 Client Ports

There are no internal or external client ports defined for this component assembly.

4.3 Publisher Ports

4.3.1 numArrayPub Publisher port (internal)

This port is used by the PSAT Publisher component to publish a portion of the array of 64-bit integers. Each time the timer fires, a variable is incremented and that variable (along with the component attribute value) become the "seed" of the numbers published. This ensures that every attachment sent is both different from one published any other publisher instance and is monotonically increasing.

4.4 Subscriber Ports

A matching subscriber port for the publisher port described in the prior section is reflectively available on the opposite monolithic component instances.

5 Component Functionality

At startup, each of the PSAT Publisher component instances set an SNA timer (per the SNA Time Management API) to generate a timer event (by default every 5 seconds). Upon timer expiration, the publisher component's timer callback will publish a PSAT attachment. When a published sample message is received by the Basic Subscriber component, it will log a message (per the SNA Logging API) to note the event. Once started, this scenario will execute indefinitely, or until the deployment is halted manually.

6 Configurable Parameters

A **PSAT_Example.cfg** configuration file allows configuration of several parameters:

Name	Component	Default Value	Description
initialDelay	PSAT_Publisher	5 secs	For the send message timer that is used to trigger the publication of a PSAT message, this is the delay before that timer triggers for the first time. *Specifiable in seconds and microseconds*
repeat	PSAT_Publisher	5 secs	For the send message timer, this is the time between all subsequent (2 .. N) timer triggers. *Specifiable in seconds and microseconds*
sendMsgLoggingModulo	PSAT_Publisher	1	When publishing messages rapidly, logging every message may be too much output. This parameter allows only every "nth" message published to be logged where "n" is this parameter value.
receiveMsgLoggingModulo	PSAT_Subscriber	2	Only every "nth" set of "m" messages received will be logged. This value is "n".

receiveMsgLoggingModuloExtent	PSAT_Subscriber	2	This value is “m” in the above description.

Since this component assembly is intended to support run-time experimentation, it is also packaged with a full set of SNA compliant run-time execution configuration and deployment files.

7 Design Constraints

1. The example assembly will implement the many-to-many attachment transfer message exchange pattern.
2. A full set of SNA compliant configuration and deployment files will be provided with this example in order to support run-time execution in a default single-host target deployment.
3. This example will utilize the standard SNA APIs to perform all functions.

8 Component Test

This 2-component assembly with two instances of each component must be executable on a single “localhost” development computer, to include the SNA SDK x86-64 VM at a minimum. Users can experiment with the deployment plan to redeploy to an alternative 2-host target environment if desired. A 2-node version of the DDS QoS file (USER_QOS_PROFILES.xml) is located in the **Deployment/scripts/twoNodes** directory of the example.

9 Component Dependencies

The Publish-Subscribe Attachment Transfer Example assembly is self contained and has no dependencies on any other application components. Its only dependency is on the SNA SDK’s run time execution environment.