



# THE JOINT ARCHITECTURE FOR UNMANNED SYSTEMS

## **Dynamic Configuration Control Document**

Version 1.2  
September 27, 2005

*This document contains information specific to the conduct of an interoperability experiment and as such does not constitute a modification or recommendation for modification to the current JAUS specification. Use of the information contained herein is done so at the user's risk. The JAUS Working Group makes no guarantee that any information contained in this document will be incorporated into the standard.*

Introduction.....	3
Dynamic Configuration .....	3
Discovery .....	3
Code D2A4h: Query Identification.....	4
Code D4A5h: Report Identification.....	5
Code D2A6h: Query Configuration.....	6
Code D4A6h: Report Configuration.....	6
Identifier Assignment.....	7
Capability Publication.....	7
Code D2A7h: Query Services.....	7
Code D4A7h: Report Services.....	7

## Introduction

J AUS defines a system in terms of subsystems, nodes, and components. The component provides a specific capability and is the main building block of the system. A node is defined to be a set of related components, and a subsystem is defined to be a set of related nodes. During system operation, components work together for the purpose of achieving a common goal. Before a component can make use of any other components' capabilities, it must be able to uniquely identify each component along with its specific capabilities. This information can be set up statically before the system is started, or it can be set up dynamically after the system is started. Dynamic Configuration is a process defined in this document that provides a mechanism for setting up a J AUS system, or part of a J AUS system, when needed rather than in advance.

## Dynamic Configuration

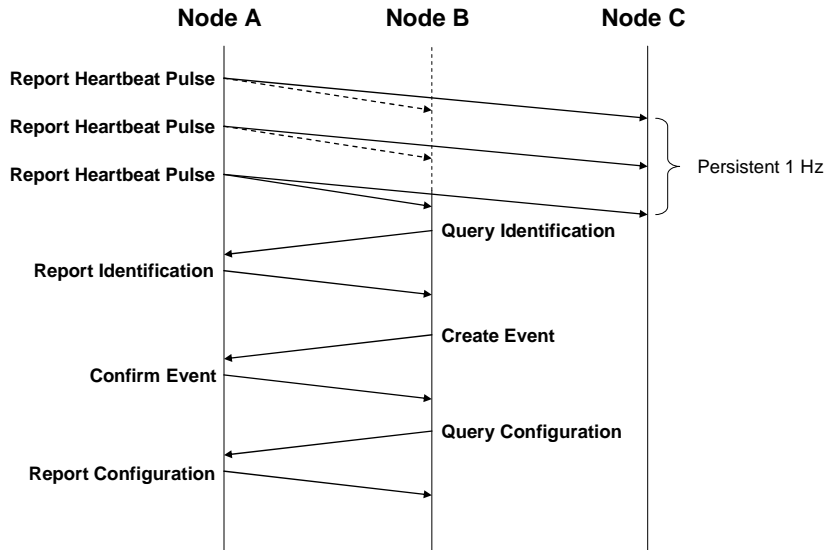
The dynamic configuration process can be broken down into the following: discovery, identifier allocation, and capability publication. Discovery is the process of determining the existence of other components, nodes, and subsystems within the system. Identifier allocation is the process of assigning unique identifiers for each component, node, and subsystem within the system. And, capability publication is the process of conveying a component's capability to any other component within the system.

### *Discovery*

The process of discovery is conducted at both the node level and the subsystem level. At the node level, a Report Heartbeat Pulse message must be sent to all nodes within the subsystem. This message must be persistent at a 1 Hz rate from all nodes within the subsystem. The J AUS destination identifier must be set to X:255:1:1, where X is the subsystem identifier. The J AUS source identifier of the Report Heartbeat Pulse message must be set to the component's identifier that is capable of responding to the messages Query Identification and Query Configuration at the node level. (Note that this is different from previous experiments where the source identifier was specified to be X:1:32:1, i.e., the subsystem commander) When a Report Heartbeat Pulse message is received by another node within the subsystem, the receiving node may query for the node's identification and configuration information. Figure 1 shows a typical flow of messages for discovering the configuration of a new node, where nodes A, B, and C are all on the same subsystem. Note that in order to keep the configuration up to date, a Create Event message can be used with the Report Configuration message. This is in contrast to the prior approach of using specific Configuration Event Setup and Notify messages.

The subsystem level discovery is very similar to the node level discovery. A Report Heartbeat Pulse message must be sent to all nodes within the system, again at a 1Hz rate. The J AUS destination identifier must be set to 255:255:1:1, and the J AUS source identifier must be set to the component's identifier that is capable of responding to the messages Query Identification and Query Configuration at the subsystem level. (Note that this is different from previous experiments where the source was specified to be X:Y:35:1, i.e., the communicator) When a Report Heartbeat Pulse message is received by a node from a different subsystem, the receiving node may query for the subsystem's

identification and configuration information. Figure 1 can be used again to show the typical flow of messages for discovering the configuration of a new subsystem and keeping it current, where nodes A, B, and C are all on different subsystems. Note again that in order to keep the configuration up to date, a Create Event message can be used with the Report Configuration message.



**Figure 1. Configuration Discovery Message Flow**

**Code D2A4h: Query Identification**

This message shall request the identification summary of a Subsystem, Node, or Component.

Field #	Name	Type	Units	Interpretation
1	Query Type	Byte	N/A	0: Reserved 1: System Identification 2: SS Identification 3: Node Identification 4: Component Identification 5 – 255: Reserved

## Code D4A5h: Report Identification

This message shall provide the requesting component an identification summary of the Subsystem, Node, or Component.

Field #	Name	Type	Units	Interpretation
1	Query Type	Byte	N/A	<p>Used to mark the type of identification being returned. Should correspond to it's respective Query.</p> <p>0: Reserved            1: System Identification            2: SS Identification            3: Node Identification            4: Component Identification            5 – 255: Reserved</p>
2	Authority	Byte	N/A	<p>Lowest level of authority required to gain control of Subsystem, Node, or Component Capabilities</p>
3	Type	Unsigned Short	N/A	<p>This field identifies the particular Unmanned Vehicle Type, Node Type or Component Type. The following values apply:</p> <p>00000 = Reserved            10001 = Vehicle TBD            10002 = Vehicle TBD            ...            20000 = Reserved            20001 = OCU TBD            20002 = OCU TBD            ...            30000 = Reserved            30001 = Other Subsystem TBD            30002 = Other Subsystem TBD            ...            40000 = Reserved            40001 = Node TBD            40002 = Node TBD            ...            50000 = Reserved            50001 = Payload TBD            50002 = Payload TBD            ...            60000– 65535 Reserved</p>

4	Identification	Byte	N/A	Human-recognizable name of the Subsystem, Node or Component. This shall be a null terminated ASCII string.
---	----------------	------	-----	--

### Code D2A6h: Query Configuration

This message shall request the configuration summary of a subsystem or node. For example, to get the complete configuration of a subsystem, field 1 shall be set to 2.

Field #	Name	Type	Units	Interpretation
1	Query Field	Byte	N/A	0: Reserved 1: Reserved 2: Subsystem Configuration 3: Node Configuration 4 – 255: Reserved

### Code D4A6h: Report Configuration

This message shall provide the receiving component a table of all existing components located on the source's subsystem or node depending on the value of field 1 of the Query Configuration message.

Field #	Name	Type	Units	Interpretation
1	Node Count	Byte	N/A	# of Nodes reported. For a single Node Report this field shall be 1.
2	NodeID1	Byte	N/A	Node ID of first node in table. For single Node or Component reporting this field shall contain the Node ID as specified in the Destination Address of the Query Configuration message
3	Comp Count 1	Byte	N/A	# of Comps reported for NodeID1
4	CompID1	Byte	N/A	Comp ID of first Component reported. For Single Component reporting this field shall contain the Component ID as specified in the Destination Address of the Query Configuration message and shall be the only Component reported
5	InstID1	Byte	N/A	Inst ID of first Component reported.
	...			
	CompIDi	Byte	N/A	Comp ID of ith comp on first node in table
	InstIDi	Byte	N/A	Inst ID of ith comp on first node in table
	NodeID2	Byte	N/A	Node ID of second node in table
	Comp Count 2	Byte	N/A	# of Comps on second node in table

	CompID1	Byte	N/A	Comp ID of first comp on second node in table
	InstID1	Byte	N/A	Inst ID of first comp on second node in table
	...			
	CompIDj	Byte	N/A	Comp ID of jth comp on second node in table
	InstIDj	Byte	N/A	Inst ID of jth comp on second node in table
	...			
	NodeIDn	Byte	N/A	Node ID of the nth node in table
	Comp Count n	Byte	N/A	# of Comp on nth node in table
	CompID1	Byte	N/A	Comp ID of first comp on nth node in table
	InstID1	Byte	N/A	Inst ID of first comp on nth node in table
	...			
	CompIDk	Byte	N/A	Comp ID of kth comp on nth node in table
	InstIDk	Byte	N/A	Inst ID of kth comp on nth node in table

### ***Identifier Assignment***

Identifier assignment is currently not addressed in this document. Therefore, all identifiers must be statically assigned at this time.

### ***Capability Publication***

Capability publication takes place at the component level. A component's capability is defined by the service it provides and the specifics about the input and output messages required to support the service. Note that a component must still provide at least one service, but it is no longer constrained to only one service. **This effectively ends the semantic link between Component ID and function. As a bridge, for OPC 2.75, we are using an enumerated list of "Services", which are currently mapped to the list of Component IDs. The motivation for doing this is to further disambiguate addressing and functionality. For now, we are not addressing the issues of protocols in Capability Publication. However, it is expected that Services will provide a list of supported (or required) protocols for their use.**

### **Code D2A7h: Query Services**

This message shall request the summary of a component's registered services. The query shall only be used to specify a single Component report. Instances of the Query Services message shall be valid only if the Instance ID in the Destination Address is not 0 or 255. This differs from the other Dynamic Registration messages, as the replies to Query Services shall only come from the component level.

### **Code D4A7h: Report Services**

This message shall provide the receiving component a summary of all services provided by a component. **Note from Table A that service type 0 is reserved for core message**

**support. This is needed since a component is now able to support multiple services and the core messages are directed to the component as a whole. For example, the command RESUME will put a component in the Ready State and therefore all services the component supports will be ready. Note that this also means that taking control of a component takes control of all the services provided by the component.** Each service includes the messages accepted as input and those provided as outputs. The command codes and presence vectors for these messages shall be listed.

Field #	Name	Type	Units	Interpretation
1	Service count	Byte	N/A	# of services supported by the component
2	Service (1)	Unsigned Short	N/A	Service Type (See Service Type Table)
3	Input Message Count	Byte	N/A	# of input messages used to support service (1)
4	Input Message (1)	Unsigned Short	N/A	Command Code for Supported Message
5	Presence Vector (1)	Unsigned Integer	N/A	Presence Vector for Command Code in previous field. This field, and all subsequent Presence Vector fields, shall always be 32 bits. For Presence Vectors smaller than 32 bits, the representative data shall be inserted with matching bit significance.
	...			
	Input Message (i)	Unsigned Short	N/A	Command Code for Supported Message
	Presence Vector (i)	Unsigned Integer	N/A	See Interpretation for Field #5
	Output Message Count	Byte	N/A	# of output messages used to support service (1)
	Output Message (1)	Unsigned Short	N/A	Command Code for Supported Message
	Presence Vector (1)	Unsigned Integer	N/A	See Interpretation for Field #5

	...			
	Output Message (j)	Unsigned Short	N/A	Command Code for Supported Message
	Presence Vector (j)	Unsigned Integer	N/A	See Interpretation for Field #5
	...			
	Service (n)	Unsigned Short	N/A	Service Type (See Service Type Table)
	Input Message Count	Byte	N/A	# of input messages used to support service (n)
	Input Message (1)	Unsigned Short	N/A	Command Code for Supported Message
	Presence Vector (1)	Unsigned Integer	N/A	See Interpretation for Field #5
	...			
	Input Message (k)	Unsigned Short	N/A	Command Code for Supported Message
	Presence Vector (k)	Unsigned Integer	N/A	See Interpretation for Field #5
	Output Message Count	Byte	N/A	# of output messages used to support service (n)
	Output Message 1	Unsigned Short	N/A	Command Code for Supported Message
	Presence Vector	Unsigned Integer	N/A	See Interpretation for Field #5
	...			
	Output Message (l)	Unsigned Short	N/A	Command Code for Supported Message

	Presence Vector (1)	Unsigned Integer	N/A	See Interpretation for Field #5
--	---------------------	------------------	-----	---------------------------------

**Table A. Service Types**

Service Description	Type
<b>Core Message Support</b>	<b>0</b>
Subsystem Commander	32
Primitive Driver	33
Global Vector Driver	34
Communicator	35
	36
Visual Sensor	37
Global Pose Sensor	38
	39
System Commander	40
Local Pose Sensor	41
Velocity State Sensor	42
Reflexive Driver	43
Local Vector Driver	44
Global Waypoint Driver	45
Local Waypoint Driver	46
Global Path Segment Driver	47
Local Path Segment Driver	48
Primitive Manipulator	49
Range Sensor	50
Manipulator Joint Position Sensor	51
Manipulator Joint Velocity Sensor	52
Manipulator Joint Force/Torque Sensor	53
Manipulator Joint Positions Driver	54
Manipulator End-Effector Pose Driver	55
Manipulator Joint Velocities Driver	56

Manipulator End-Effector Velocity State Driver	57
Manipulator Joint Move Driver	58
Manipulator End-Effector Discrete Pose Driver	59

