

ICCP TASE.2 Overview

ICCP – Inter Control Center Protocol (also known as Telecontrol Application Service Element - TASE.2) has been standardized under the IEC 60870-6 specifications and allows for data exchange over WANs between a utility control center and other control centers, utilities etc.

The important specifications are

- **IEC 60870-6-503** – Defines the ICCP Application Modeling and Services
- **IEC 60870-6-702** – Defines the Application Profile for use with ICCP
- **IEC 60870-6-802** – Defines a set of standardized Object definitions

The ICCP protocol runs over the MMS (Manufacturing Message Specification – ISO/IEC 9506) protocol and the specification fully defines the mapping of ICCP Object models to MMS Object models as well as the mapping of ICCP Operations and Actions to MMS services.

Object Models, Actions and Operations

Some of the important object models and services are as below.

- **Associations**
An application Association needs to be established between two ICCP instances before any data exchange can take place. Associations can be Initiated, Concluded or Aborted by the ICCP instances
- **Bilateral Agreement and Table, Access Control**
A Bilateral Agreement between two control-centers (say A and B) is a document that defines the objects that A is willing to allow B to access and the objects that B is willing to allow A to access. A Bilateral Table is a digital representation of the Agreement. An ICCP instance can maintain many BLTs (one for each other Control Center that it interacts with)
- **Data Values**
Data Values are objects that represent the values of control-center objects including points (Analog, Digital and Controls) or data structures. ICCP defines four operations for Getting the data value, the data value name, the data value type and for Setting the data value.
- **Data Sets**
Data Sets are ordered-lists of Data Value objects that can be created locally by an ICCP server or on request by an ICCP client. ICCP defines 6 operations for Data Sets including creating and deleting Data sets, getting the Data set element values, names and types and for setting the Data Set element values
- **Information Messages**
Information Message objects are used to exchange text or other data between Control Centers. An IM object consists of a header part and an Info stream part that is not limited to printable characters
- **Transfer Sets**
Transfer Set objects are used for complex data exchange schemes to transfer Data Sets (all elements or a subset of the Data set elements) etc. This scheme allows for the data set elements to be reported periodically or based upon some other trigger conditions. It also allows for tagging a report as critical, which will prompt the receiving ICCP instance to acknowledge the receipt of the report
- **Devices**
Devices are the ICCP objects that represent controllable objects in the control center. Device objects represent both Direct-Control as well as Select-Before-Operate controls and can be analog or binary controls. ICCP also allows for tagging Device objects as inoperable remotely. ICCP defines 4 operations for Select, Operate, Get Tag and Set Tag of a device object. In addition ICCP defines 4 actions for Timeout, Local reset, Success and Failure

Virtual Control Center

ICCP defines the VCC concept to represent an abstraction of a real-world control center that acts like an ICCP wrapper to the Control Center. The VCC includes all the information for representing all the control-center elements that have an ICCP context. It is modeled using the objects and services mentioned above.

A VCC wrapped around a real Control Center may have ICCP communications with many other VCCs (representing remote control centers). Each pair of interacting VCCs is governed by a Bilateral Table. Further, information in a VCC may be defined to have global scope (VCC scope) or a limited scope (ICC Scope). VCC scope expands the scope of the information to more than one control center participating in the ICCP network. ICC scope limits the information to have scope only with respect to one client control center

Data Transfer Mechanisms

ICCP implementation entails wrapping real-world control center objects in ICCP Data Value and Device objects. ICCP also allows for grouping these Data Value objects into Data Sets based upon any criteria that the implementer may wish to use to classify these objects.

DS conditions

Data objects may be setup for One-shot transfers based upon a simple Get request from a client. However ICCP typically uses more complex mechanisms involving the use of Data Sets and DataSetTransfer Sets. Here the Data sets are associated with Condition monitoring parameters that can cause the Data set contents to be sent as a Transfer Report. The Transfer Report can be triggered by a number of Conditions including object change (events), operator requests, periodic timers, Integrity poll timers etc.

DSTransmission Parameters

When a Transfer Report is triggered ICCP defines several Transmission Parameters to define what is sent. For example the RBE parameter defines whether to send all values in the Data set or just the changed values. Setting the Buffer time parameter instructs ICCP to wait for that amount of time on triggering before collecting the Data Values. This allows for cascading events to be captured and sent in just one Report. Setting the critical parameter instructs ICCP to expect an acknowledgement for that transfer.

Conformance Blocks

ICCP divides the entire ICCP functionality into 9 conformance block subsets. Implementations can declare the blocks that they provide support for, thus clearly specifying the level of ICCP supported by the implementation. Any ICCP implementation must necessarily support Block 1

Block 1 – Basic Services

- Association objects
 - Initiate
 - Conclude
 - Abort
- Data Value objects
 - Get Data Value
 - Set Data Value
 - Get Data Value Names
 - Get Data Value Type
- Data Set objects
 - Create Data Set
 - Delete Data Set

- Get Data Set Element Values
- Set Data Set Element Values
- Get Data Set Names
- Get Data Set Element Names
- Dataset-TransferSet objects
 - Start Transfer
 - Stop Transfer
 - Data Set Transfer Set Condition Monitoring
 - IntervalTimeout condition monitoring
 - OperatorRequest condition monitoring
 - (Other conditions monitoring in Block 2)
- Next Data Set Transfer Set object
 - Get Next DSTransferSet Value

Block 2 – Extended Data Set Condition Monitoring

- Data Set Transfer Set Condition Monitoring
 - ObjectChange condition monitoring
 - IntegrityTimeout condition monitoring

Block 3 – Blocked Transfers

- Transfer Reports with Block data

Block 4 – Information Message

- Information Message objects
- IMTransfer Set objects
 - Start Transfer
 - Stop Transfer
 - Data Set Transfer Set Condition Monitoring

Block 5 – Device Control

- Device objects
 - Select
 - Operate
 - Get Tag
 - Set Tag
 - Timeout
 - Local Reset
 - Success
 - Failure

Block 6 – Programs

Block 7 – Event Enrollment

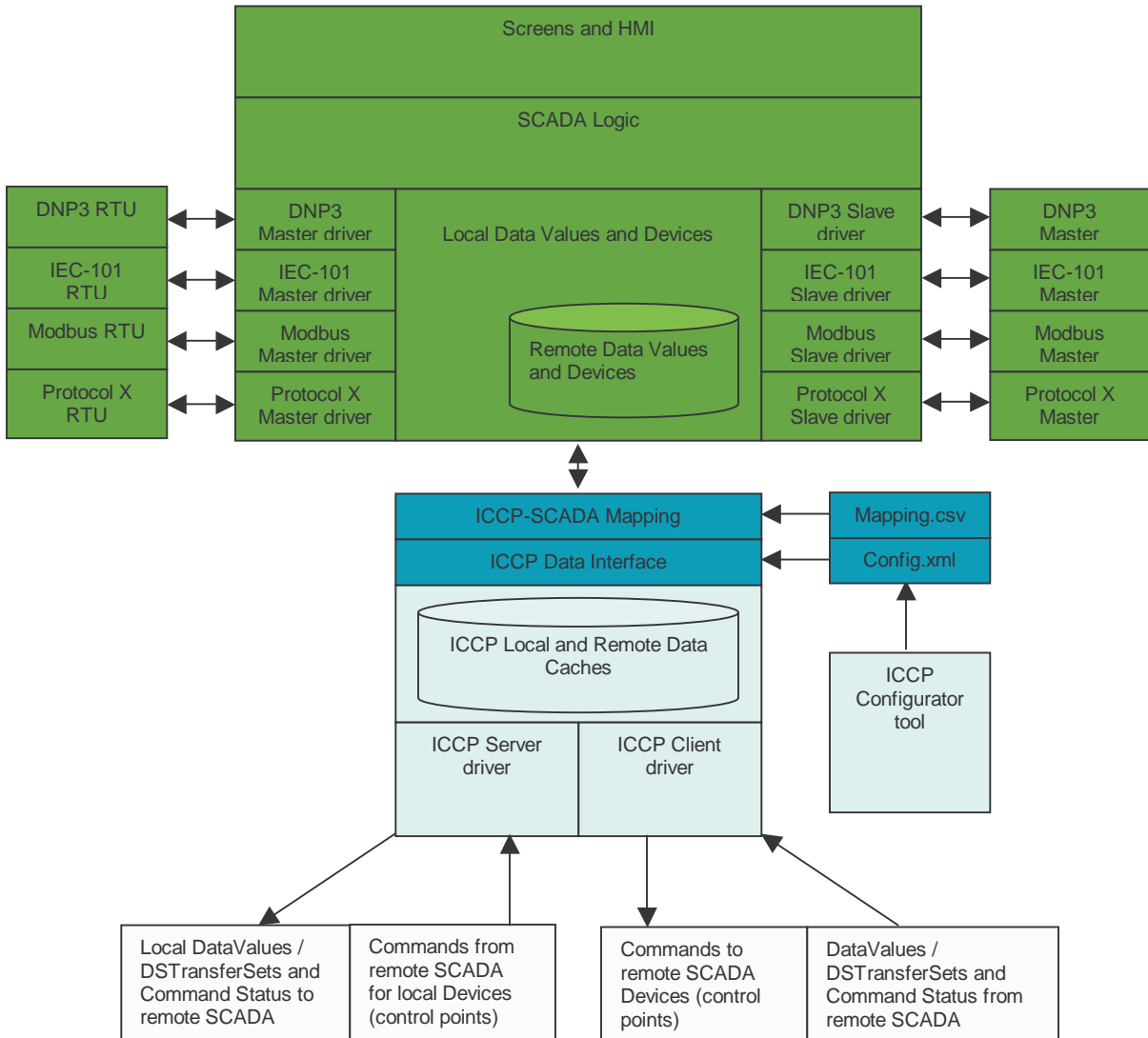
Block 8 – Accounts

Block 9 – Time Series

Kalki ICCP-TASE.2 Solutions

Kalki provides an ICCP Server with API's for integrating the same into any Control Center SCADA package seamlessly in association with our implementation services. The Server consists of the well-defined application interfaces for adding the hooks to integrate the protocol into external control center software.

The ICCP implementation is fully conforming to Blocks 1,2,4 and 5

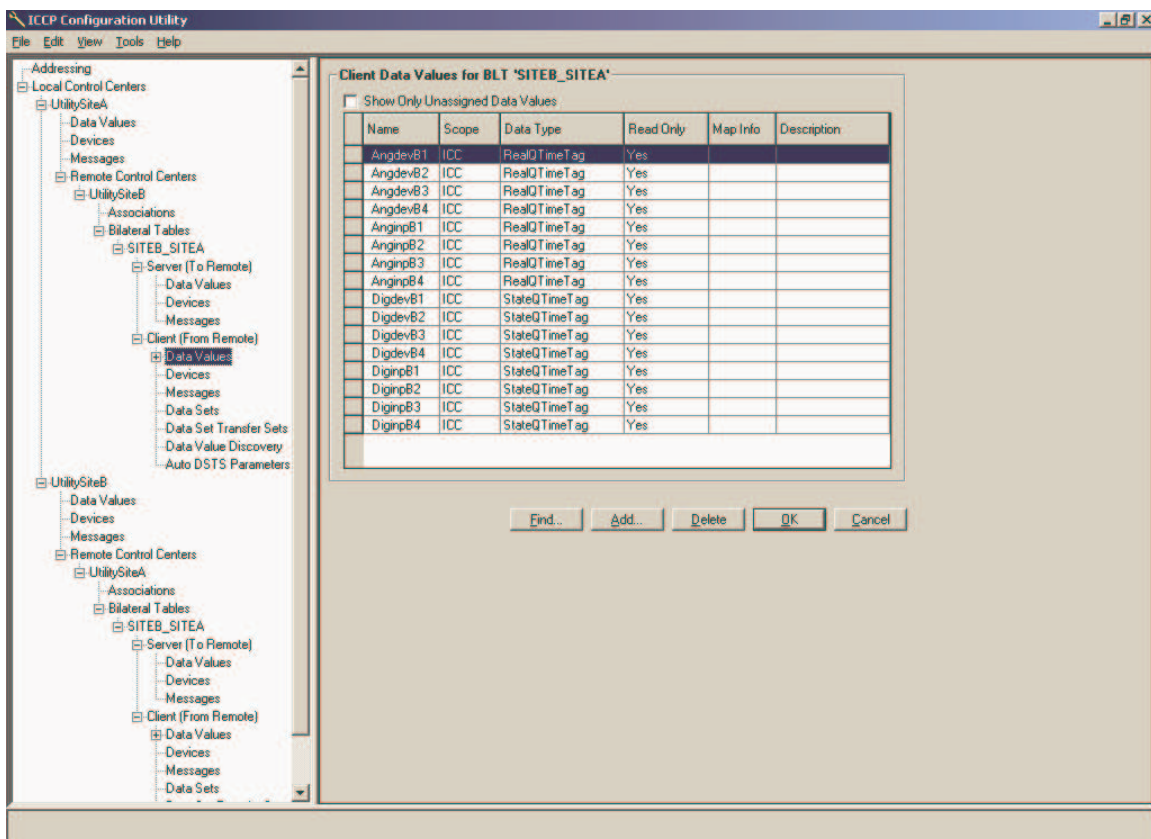


Code	Description	Scope
	SCADA engineering and logic	OEM
	Mapping and integrating data	OEM & Kalki
	ICCP SCL, libraries, tools and executables	Kalki

Configuration

The provided configuration tool allows the user to configure the Data Values and Devices for multiple SCADA nodes simultaneously.

1. Configure IP Addresses, Reference Names, Links and Titles for each SCADA node
2. Configure a VCC for each different SCADA node
3. Configure Local DataValues and Devices
4. Configure Remote Control Centres for each VCC (select from the list of other VCCs, whichever VCCs are to communicate with this one)
5. For each remote VCC to be connected with the local VCC, configure a Bilateral Table (Agreement on resource sharing and access) and Associations (multiple redundant associations are possible)
6. Within the Bilateral Table, configure a subset of the local DataValues and Devices to be exposed to the remote with access permissions (this VCC acting as an ICCP Server)
7. Within the Bilateral Table, configure the client DataValues and Devices (remote DVs and Devices) that are to be accessible to this VCC acting as an ICCP Client
8. Group local DataValues into different DataSets
9. Assign TransferSets for carrying the DataSets and set Condition-monitoring parameters which specify under what conditions will each DataSet be triggered to be packaged and sent to the remote (Conditions may be interval-timeout, integrity-timeout, event-based-triggering etc.)
10. Specify filtering conditions like RBE which specifies whether the entire DataSet needs to be sent out or only the changed values in the DataSet need to be sent out



The tool is used to export the configuration data in the form of XML files that can be directly read and processed by the ICCP executable.

Data Mapping

Mapping Data types

Under ICCP all Monitor points (called Inputs in some SCADA terminologies) are called as "DataValues". DataValues again can be classified into Analog and Digital DataValues where Analog DVs are further classified into "Real" and "Discrete" datatypes (Real stands for floating point values and Discrete stands for Integer values). Digital values are called "State"

All Control points (Called Outputs elsewhere) are called as "Devices".

Please note that the corresponding Monitor point for an Output point can also be configured as a DataValue. That means a single physical Control point (say a Breaker or a Setpoint) can have both a "Device" point as well as a "DataValue" point engineered in ICCP

The corresponding types are designated as below in ICCP

Sno	Data class	Data type	ICCP class	ICCP type	Modifiers
1	Analog Input	Floating point values	DataValue	Real	<ol style="list-style-type: none"> 1. With Quality 2. With Quality & Timestamp 3. Extended
		Integer values		Discrete	
2	Digital Input	Boolean values		State	
3	Analog Output	Floating point setpoints	Device	Real	None
		Integer setpoints		Discrete	
4	Digital Output	Boolean outputs		Command	

Mapping Data points

ICCP identifies its Data Values and Devices by a unique name. The SCADA may have its own methodology for identifying and distinguishing between points. For example, the SCADA pointid may consist of a Station number, Data class number and point index.

Each SCADA id needs to be mapped to a unique ICCP name and the same name needs to be configured into the Configuration tool.

This is an integration-phase effort and depends upon the nature of the SCADA id system. This mapping is typically performed by using comma-separated-values files, but the exact nature of the mapping system is flexible and can be modified to suit the requirements.

Data and Control Interface API

The Data and Control interface consists of the following functions that are called by the ICCP executable and callbacks that can be called by the SCADA interface. These functions need to be implemented to suit the mapping and nature of the interface to the SCADA. For SCADA interfaces that support only polling, Data Values can be updated by using the "Get" functions. For SCADA interfaces that support event-based triggering, the "Update" callbacks can be called from the interface

Initialization and Exit

InitDataInterface()

This function is called by the ICCP executable to initialize the Data interface. This function needs to be implemented to handle activities like opening connections, opening file handles etc. which are required to start-up the interface

ExitDataInterface()

This function is called by the ICCP executable to close the data interface. This will usually happen when the ICCP executable detects a connection failure (connection to the local SCADA fails), wherein it will close and try to re-initialize the interface

UpdateAssociationStatus()

This function is called by the ICCP executable to inform the SCADA about Association establishment and failure. (Associations to remote SCADAs)

ICCP Server

These functions are called repeatedly by the ICCP executable to update its cache of data with the latest values from the SCADA

GetDataValueAndQuality()

GetDeviceStatus()

GetStationStatus()

These functions are called by the ICCP executable when it receives requests from a remote ICCP client

SelectDevice()

OperateDevice()

These callbacks can be called by the SCADA interface asynchronously to update the ICCP server's cache

Callback UpdateDataValueAndQuality()

Callback UpdateDeviceStatus()

Callback UpdateStationStatus()

Callback UpdateCommandStatus()

ICCP Client

The following functions are called by the ICCP client when it receives data from a remote server

StoreDataValueAndQuality()

StoreDeviceStatus()

StoreStationStatus()

The following callbacks can be called by the local SCADA when it requires to send commands or requests to a remote server

Callback RequestSelectDevice()

Callback RequestOperateDevice()

Callback RequestDataValueAndQuality()

The following callbacks can be called by the local SCADA when it wants to refresh information in its database from the ICCP client's cache. These calls do not cause a request to go out to the remote server. The data is supplied from the ICCP client's cache locally

Callback RefreshDataValueAndQuality()

Callback RefreshDeviceStatus()

Callback RefreshStationStatus()

