I-TECHNOLOGY INC.

System Monitor Specification

Confidential

June 6, 2019
Revision 2.1

Copyright

I-Technology Inc.
System Monitor Specification

Document Revision History

Version	Date	Description of Change	Person Responsible
1.0	2014/01/14	Initial Release	Bob Jackson
1.1	2014/01/15	1. Modified application screens 2. Changed Start & Stop System messages 3. Modified Data Definition fields 4. Added device name and application name to various system messages	Bob Jackson
1.2	2014-01-17	1. Deleted Version Request and Version Data System Messages. They will be handled by Ping & Ping Response	Bob Jackson
1.3	2014-01-29	1. Added Grid and Button descriptions to System Monitor Web Application screen descriptions. 2. Added Parameters Set System Message.	Bob Jackson
1.4	2016-02-09	1. Removed Subscription Requests and Subscription Grants	Bob Jackson
1.5	2017-03-01	1. Add Query Report	Bob Jackson
1.6	2017-03-10	1. Modify ErrorAction 2. Remove Subscription Request and associated Subscription List Data Records.	Bob Jackson
1.7	2017-04-07	Modify data definitions	Bob Jackson
1.8	2017-04-26	Add 9000005, 9000006, 9000007, 9000070, 9000071 Data Types	Bob Jackson, Jamie Rogers
1.9	2017-10-05	Update Start record to add Subscriptions and Publications	Bob Jackson
2.0	2018-12-13	Added Notice of Failure System Message	Bob Jackson
2.1	2019-06006	Added PID to starting, stopping and ping response records	Bob Jackson

Table of Contents

1 Introduction

A System Monitor is a web application that participates in an I-Technology Inc. Central Data environment. It acts to handle System Messages and provide oversight on the operation of the components of the system.

1.1 Application Operation

Central Data Applications participate in a publish and subscribe regime whereby they publish data to a wider audience and subscribe to data coming from other participants. In aid of this, the application needs a Publisher object that will establish a connection with a Data Broker and transmit the data to be published.

The application also requires a Subscriber Task that establishes a socket connection to the Broker, informs it of the data types that it is subscribing to and watches the socket for any published data. When data is received, the subscriber passes it on in an inter-task queue.

The System Monitor is a Python application that stores and displays operating information.

System messages are described in Appendix A and B. Some generated by applications and subscribed to by the System Monitor. Some are generated by the System Monitor to elicit responses from applications.

The System Monitor provides a user interface to allow the user to monitor various aspects of the systems operation and to effect changes.

1.1.1 System Monitor Functions

The System Monitor will be a web application deployed on an Apache Tomcat server. It will perform the following functions:

1. Monitor the status of applications in the system.
2. Keep a record of and provide when required, the parameters for each application.
3. Monitor error messages from applications and generate alerts.
4. Provide debug logging information from applications.
5. Synchronize applications by acting as a Time Server.
6. Keep track of the versions of the applications running in the system.

These functions are made possible by the System Messages. Event Types (9000000.00 – 9999999.99) are reserved for System Data. These events are defined in the table below. Those relevant to the System Monitor are marked with an asterisk

System Events

Type	Name	Issuing Event
9000000.00*	Started	Issued when an application has started
9000001.00*	Stopping	Issued when an application is shutting down
9000010.00*	Ping	Issued by the System Monitor to determine if an application is still operating.
9000011.00*	Ping Response	Response from the application being pinged.
9000020.00*	Error	Used to alert about faults an application has experienced.
9000025.00	Notice of Failure	Issued by a dependant micro-service when an expected response is not received.
9000030.00*	Status	Used to send a global message to all applications.
9000040.00*	Time	A Timestamp used to synchronize an application with other applications in the system. Issued by the System Monitor on request.
9000041.00*	Time Request	Issued by an application when it needs to synchronize with the system.
9000060.00*	Parameter Request	Issued by an application requesting it's parameters from the System Monitor
9000061.00*	Parameters	Sent by the System Monitor in response to a Parameter Request
9000080.00*	Data Definition	A Subscriber when connecting with a Broker. The System Monitor in response to a Version Request. The data will be the XML documents that describe each version of the data type in the Version Request
9000085.00	Query Report	Published by the Librarian when it receives a query

Note:

Appendix I will detail the syntax of these various messages.

1.1.1.1 System Monitor Application

1.1.1.1.1 Log On

The Log on screen is shown in the markup below.

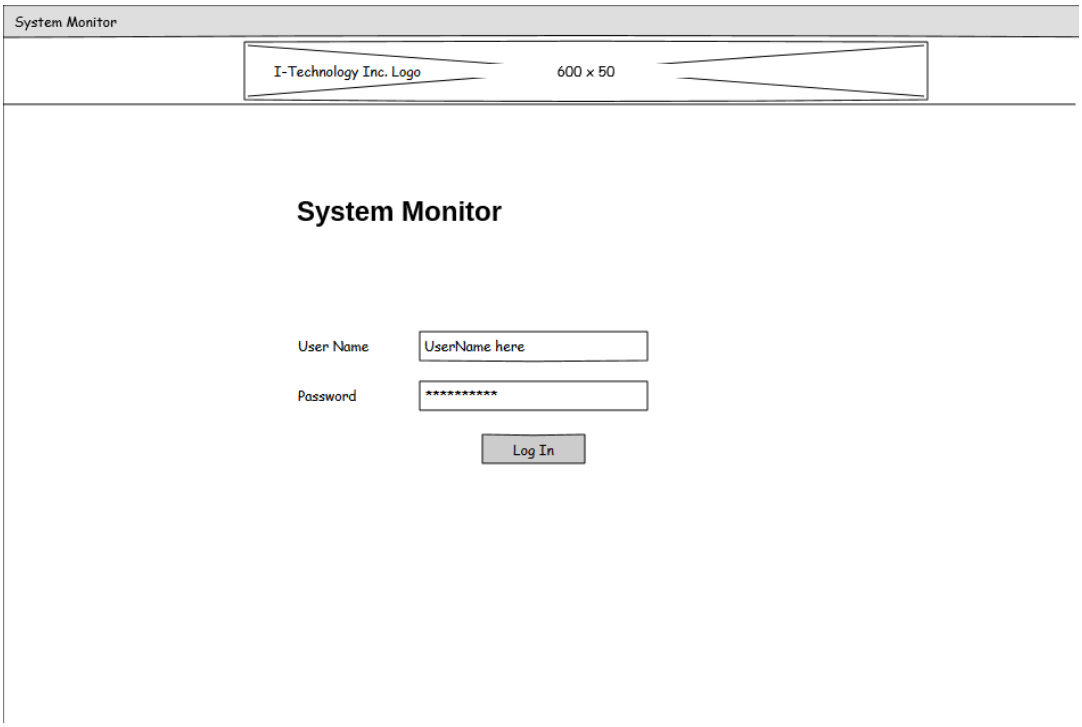
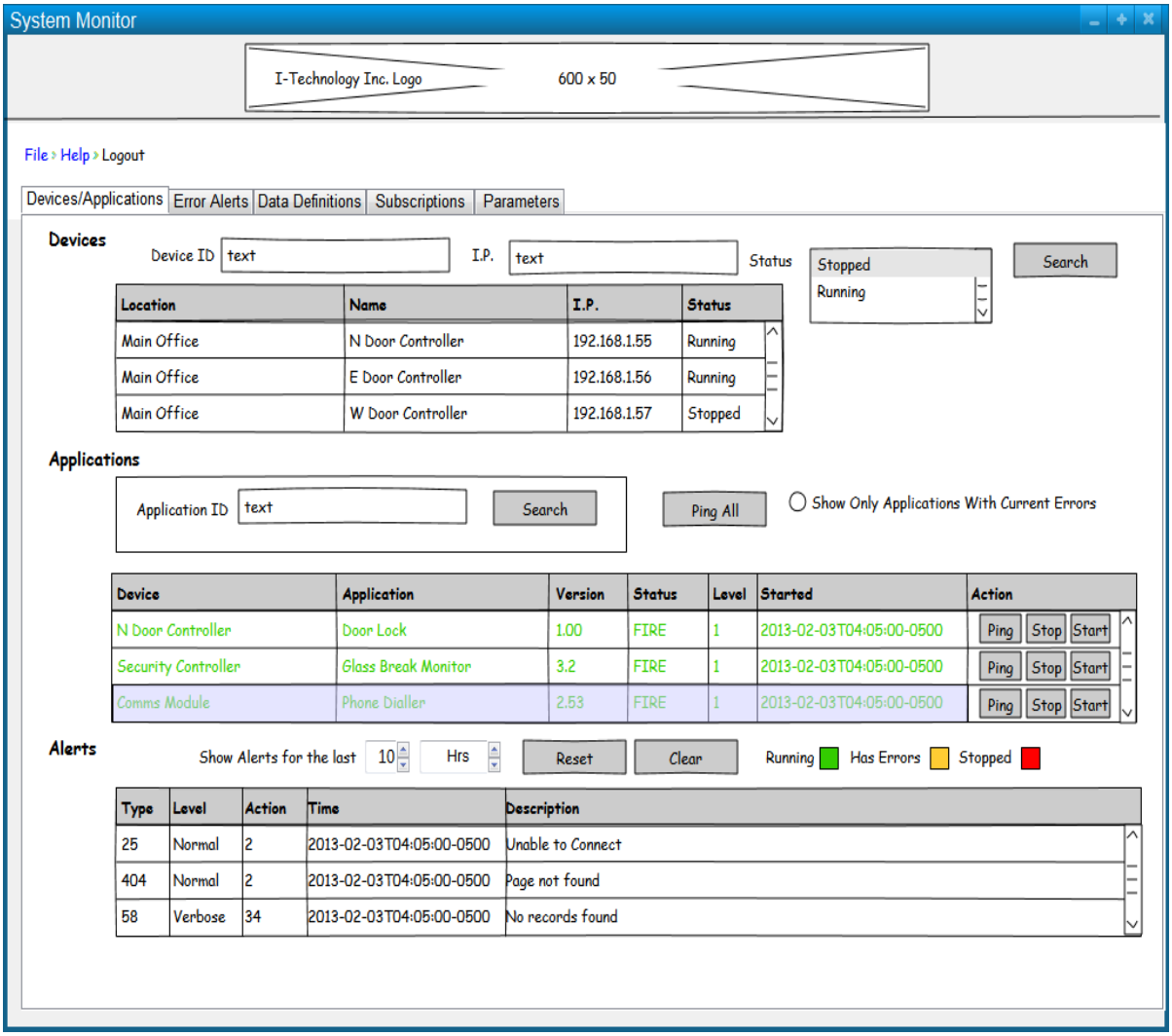


Illustration 1: System Monitor Log On Screen

1.1.1.1.2 Devices /Applications Monitoring

Devices and Applications are handled on the Devices/Applications Screen.

Illustration 2: System Monitor Devices & Applications



Grids:

1. **Devices** – A table of devices compiled from deviceName and location fields in Started [9000000.00] or Ping Response [9000011.00] messages or Ping Response [9000011.00].
2. **Applications** – A table of applications compiled from deviceName and application Name fields in Started [9000000.00] or Ping Response [9000011.00] messages or Ping Response [9000011.00].
3. **Alerts** – From a table of alert messages from Error [9000020.00] messages.

Buttons:

1. Search – Filters the associated grid based on the contents of the text fields.
2. Ping – Publishes a Ping [9000010.00] message addressed to the device and application.
3. Ping All - Publishes a Ping [9000010.00] message addressed to the all the known devices and

applications.

4. Reset – Resets the Alerts grid filter to show only new messages.
5. Clear – Clears the Alerts grid filter. (Shows all alerts).

1.1.1.1.3 Parameter Inventory

The Set Parameters and Get Parameters buttons on the Devices and Applications tab will handle the parameters for the selected application.

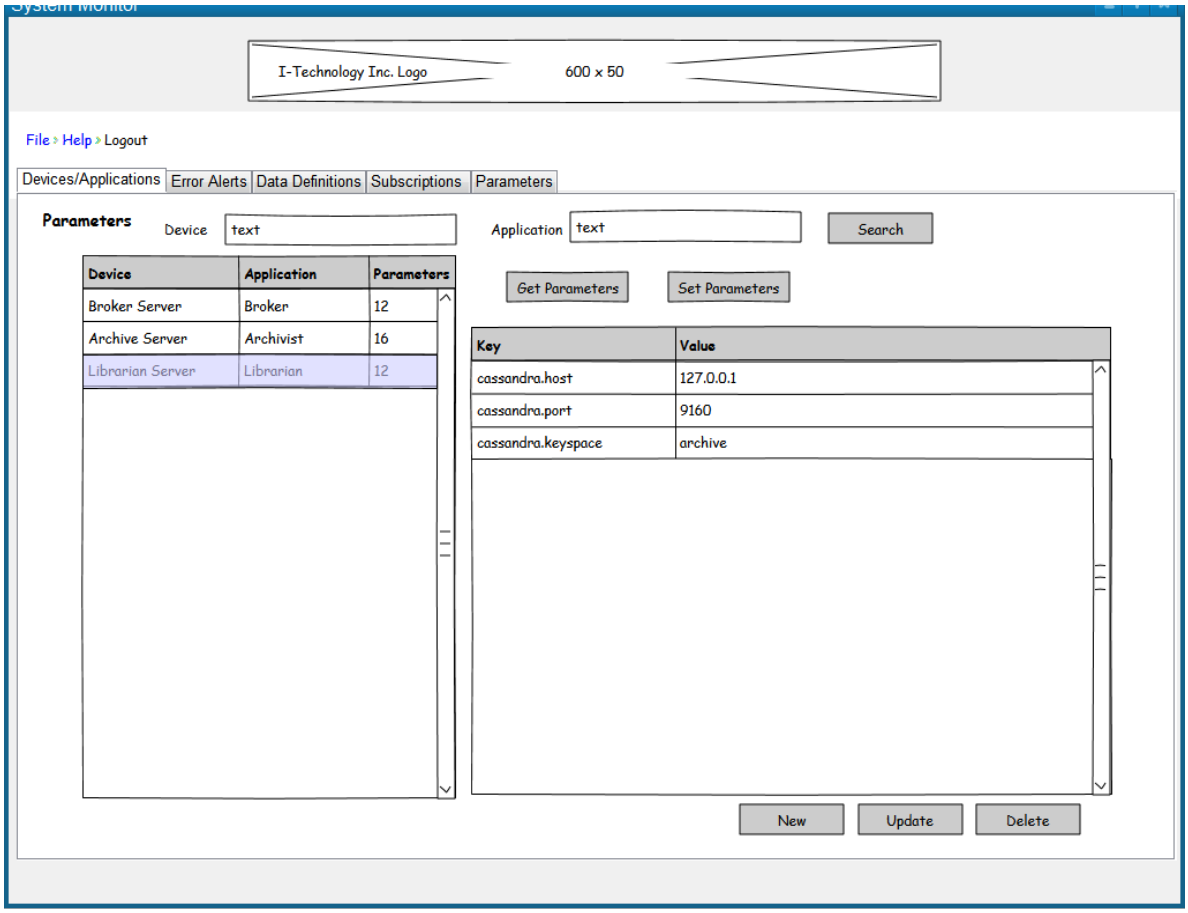


Illustration 3: System Monitor - Parameters

Grids:

1. **Parameters** – A table of parameters compiled from the fields in Parameters [9000061.00] messages.
2. **Parameter Values** – A table of parameter values compiled from the fields in Parameters [9000061.00] messages. (Linked to the table above)
3. **Alerts** – From a table of alert messages from Error [9000020.00] messages.

Buttons:

1. Search – Filters the associated grid based on the contents of the text fields.

2. Get Parameters – Publishes a Parameter Request [9000060.00] message addressed to the device and application.
3. Set Parameters - Publishes a Parameter Set [9000062.00] message addressed to the device and application based on the current values in the parameters and parameter values tables.
4. New – Creates a new record in the associated grid.
5. Update – Changes data in the selected row of the associated grid..
6. Delete – Deletes the selected parameter or parameter value from the associated grid.

1.1.1.1.4 Error Reporting

A library will be provided similar to log4j. It will send all log messages defined in code by the programmer out as a system message [9000020.00]. These messages are subscribed to by the System monitor and handled by the error logger.

Errors will be handled on the Error Alerts Page.

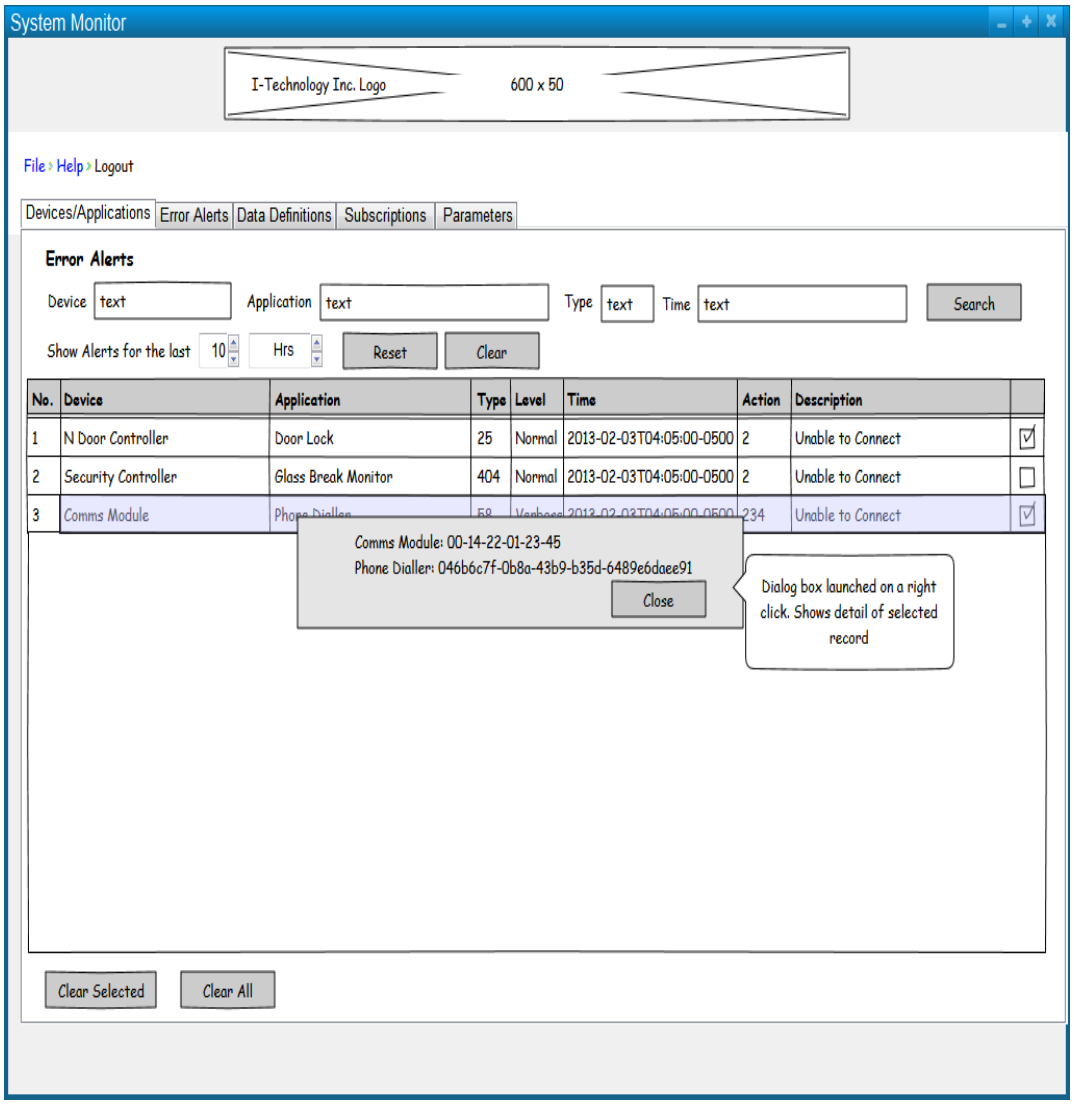


Illustration 4: System Monitor Error Alerts

Grids:

1. **Error Alerts** – From a table of alert messages from Error [9000020.00] messages.

Buttons:

1. Search – Filters the associated grid based on the contents of the text fields.
2. Reset – Resets the Alerts grid filter to show only new messages.
3. Clear – Clears the Alerts grid filter. (Shows all alerts).
4. Clear Selected – Clears those messages selected using the check-boxes.
5. Clear All – Clears all messaages from the grid.

1.1.1.1.5 Debug/Trace Monitor

Debug messages will be sent to the System Monitor if the application status is set to Verbose. These messages will appear on the Error Alert tab.

1.1.1.1.6 Time Server

The System Monitor will respond to Time Request message by replying with a Time message. Systems will be time synchronized with an error proportionate to the transaction delay.

1.1.1.1.7 Data Definitions

Data Definitions are catalogued on the Data Definitions Tab.

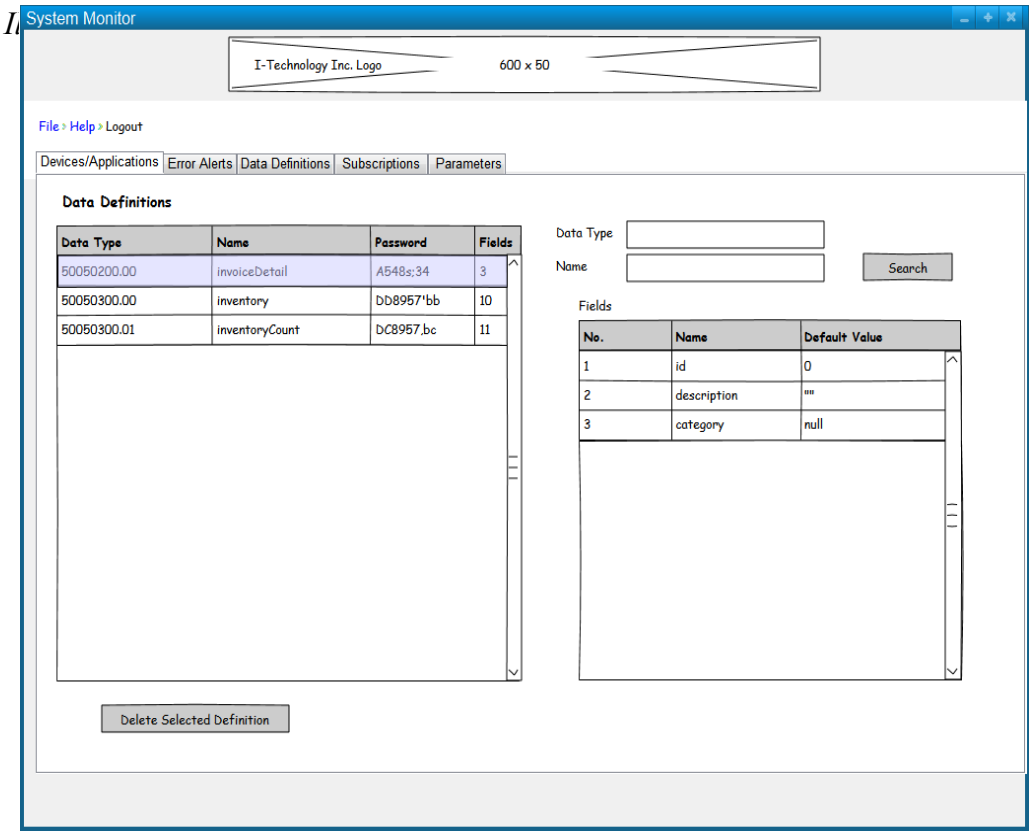


Illustration 6: System Monitor - Data Definitions

Grids:

1. **Data Definitions** – From a table of Data Definitions from Data Definition [9000080.00] messages.
2. **Data Definition Field Values** - From a table of Data Definition Field Values from Data Definition [9000080.00] messages. Shows those fields associated with the selected Data Definition.

Buttons:

1. Search – Filters the associated grid based on the contents of the text fields.
2. Delete Selected Definition – Deletes the selected data definition from the table.
3. Clear All – Clears all messaages from the grid.

1.1.1.1.8 Subscription Monitor

Subscriptions are tracked on the Subscriptions tab.

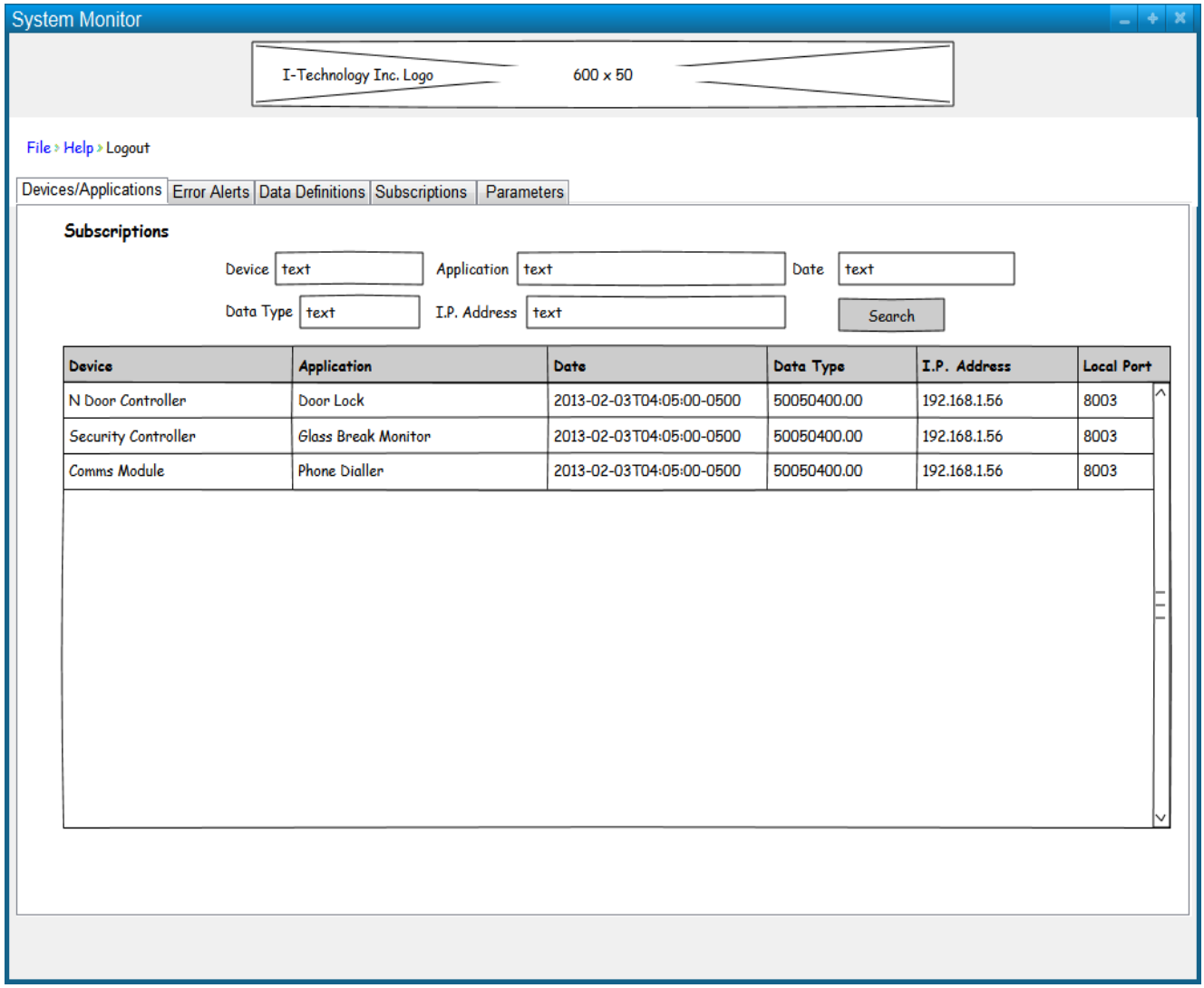


Illustration 7: System Monitor Subscriptions Tab

Grids:

1. **Subscriptions** – From a table of Subscriptions from Subscription List [9000056.00] messages received in response to System List Request [9000055.00] messages.

Buttons:

1. Search – Filters the associated grid based on the contents of the text fields.

Appendix A XML Message Formats

A 1. Publication

Each publisher is identified by it's socket information (IP Address and Port). If multiple applications are deployed (on a web server for example) the developer must make each one unique by using different ports.

A.1.1 Publication DTD

The DTD below describes the xml for a publication:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE packet[
<!ELEMENT packet (publisherId, metadata, dataRecord)>
<!ELEMENT publisherId (date, ipAddr, localPortNo)>
<!ELEMENT metadata (recordNo, type, date, status, creator, link, tenant, fieldCount)>
<!ELEMENT recordNo (#PCDATA)>
<!ELEMENT type (#PCDATA)>
<!ELEMENT date (#PCDATA)>
<!ELEMENT status (#PCDATA)>
<!ELEMENT creator (#PCDATA)>
<!ELEMENT link (#PCDATA)>
<!ELEMENT tenant (#PCDATA)>
<!ELEMENT fieldCount (#PCDATA)>
<!ELEMENT dataRecord (field+)>
<!ELEMENT field (fieldNo, data)>
<!ELEMENT fieldNo (#PCDATA)>
<!ELEMENT data (#PCDATA)>
<!--The IP address of the source-->
<!ELEMENT ipAddr (#PCDATA)>
<!--The local port number  of the source-->
<!ELEMENT localPortNo (#PCDATA)>
]>
```

example:

```
<packet>
      <publisherId>
            <date> 2013-02-03T04:05:00-0500 </date>
            <ipAddr>192.168.1.68</ipAddr>
            <localPortNo>4003</localPortNo>
      </publisherId>
      <metadata>
            <recordNo>dfe010d0-e805-11e2-8ba9-68b599e2e914</recordNo>
            <type>50050200.00</type>
            <date>2011-02-03T04:05:00-0500</date>
            <status>0</status>
            <creator>”DemoUser”</creator>
            <link>0</link>
            <tenant>5</tenant>
            <fieldCount>3</fieldCount>
      </metadata>
      <dataRecord>
            <field>
                  <fieldNo>1</fieldNo>
                  <data>”12345”</data>
            </field>
            <field>
                  <fieldNo>2</fieldNo>
                  <data>”Smith”</data>
            </field>
            <field>
                  <fieldNo>3</fieldNo>
                  <data>”Robert”</data>
            </field>
      </dataRecord>
</packet>
```

Appendix B Librarian Queries

The following example details a query to the Librarian.

```
<9000010.00>            # Query for data records of type  9000010.00

<0>                     # Query for all records in the Archive
```

Proposal:

```
 <applicationId>
<applicationName>
<deviceId><
deviceName>
<userName>
```

<tenantId>
<74beb9bf-b314-458f-bdbe-bc5e2dfede28>

Appendix C System Data Messages

System Data Messages are created in the same format as Publications which are detailed in Section 1 above. The tables below detail the fields in the various System Data Messages. The identity of an application will be determined using two fields:

1. The MAC address of the application's host
2. A UUID generated by the application developer that will uniquely identify the application

The application may have several implementations on different machines and a web server may host several applications.

C.1.1 Started [9000000.00]

Field #	Field Name	Data Type	Description
1	deviceID	String	Mac Address
2	deviceName	String	Device name
3	location	String	Device Location
4	applicationID	byte(16)	UUID assigned by developer
5	applicationName	String	The name of the application
6	StartTime	Timestamp	Local Start Date/Time
7	Subscriptions	String	List of records subscribed to
8	Publications	String	List of records published
9	pid	Integer	The Process ID

C.1.2 Stopping [9000001.00]

Field #	Field Name	Data Type	Description
1	deviceID	String	Mac Address
2	deviceName	String	Device name
3	Location	String	Device Location
4	applicationID	byte(16)	UUID assigned by developer
5	applicationName	String	The name of the application
6	stopTime	Timestamp	Local Stop Date/Time
7	pid	Integer	The Process ID

C.1.3 Power Loss [9000005.00]

Field #	Field Name	Data Type	Description
1	vin	float	Voltage In
2	vbat	float	Battery Voltage
3	vout	float	Voltage Out

C.1.4 Power Restored [9000006.00]

Field #	Field Name	Data Type	Description
1	vin	float	Voltage In
2	vbat	float	Battery Voltage
3	vout	float	Voltage Out

C.1.5 Powering Off [9000007.00]

Field #	Field Name	Data Type	Description
1	vin	float	Voltage In
2	vbat	float	Battery Voltage
3	vout	float	Voltage Out

C.1.6 Ping [9000010.00]

Field #	Field Name	Data Type	Description
1	pingType	Integer	0 = all, 1 = device, 2 = application
2	deviceID	String	Mac Address
3	applicationID	byte(16)	UUID assigned by developer
4			

C.1.7 Ping Response [9000011.00]

Field #	Field Name	Data Type	Description
1	deviceID	String	Mac Address
2	deviceName	String	Device Name
3	location	String	Device Location
3	applicationID	byte(16)	UUID assigned by developer
4	applicationName	String	The name of the application
5	pingTime	timestamp	UTC Date/Time of response
6	pid	Integer	The Process ID

C.1.8 Error [9000020.00]

Field #	Field Name	Data Type	Description
1	deviceID	String	Mac Address
2	deviceName	String	Device Name
3	applicationID	byte(16)	UUID assigned by developer
4	applicationName	String	The name of the application
5	errorType	Integer	Error Code defined by Developer
6	errorLevel	Integer	Error Level for Verbosity filtering
7	ErrorAction	Integer	Action code: 0 – Display 1 – Email Alert Level 1 2 – Email Alert Level 2 3 – Email Alert Level 3 4 – Page Alert 5 – Syslog Alert
8	errorText	String	Description of the Error

C.1.9 Notice of Failure [9000025.00]

Field #	Field Name	Data Type	Description

1.	Message I.D.	UUID	I.D. of the initiating message
2.	Session I.D.	UUID	Session Identifier
3.	Response List	List	Expected Response(s)
4.	Responding Service	UUID	Application I.D. of responding service
5.	Initiating Message	String	The message not responded to

C.1.10 Status [9000030.00]

Field #	Field Name	Data Type	Description
1	deviceID	String	Mac Address
2	applicationID	byte(16)	UUID assigned by developer
3	StatusLevel	Integer	Priority – 10 = highest
4	StatusNumber	Integer	A unique number identifying the kind of status.
5	StatusMessage	String	Text of the Status Message

C.1.11 Time [9000040.00]

Field #	Field Name	Data Type	Description
1	deviceID	String	Mac Address
2	applicationID	byte(16)	UUID assigned by developer
3	Time	Timestamp	System Time for synchronization
4			

C.1.12 Time Request [9000041.00]

Field #	Field Name	Data Type	Description
1	deviceID	String	Mac Address
2	applicationID	byte(16)	UUID assigned by developer
3			
4			

C.1.13 Parameter Request [9000060.00]

Field #	Field Name	Data Type	Description
1	deviceID	String	Mac Address
2	applicationID	byte(16)	UUID assigned by developer
3			
4			

C.1.14 Parameters [9000061.00]

Field #	Field Name	Data Type	Description
1	deviceID	String	Mac Address
2	deviceName	String	Device Name
3	applicationID	byte(16)	UUID assigned by developer
4	applicationName	String	The name of the application
5	parameterCount	Integer	# of Parameters
6	Parameter* parameterKey parameterValue	 String String	 Key name of Parameter Parameter Value

* Repeating Key-Value Pair

C.1.15 Configuration Update/Read File [9000070.00]

Field #	Field Name	Data Type	Description
	deviceId	String	Mac Address
	deviceName	String	Device Name
	location	String	Device Location as set by installer
	applicationId	byte(16)	UUID assigned by developer
1	applicationName	String	The name of the application
2	authorization	Integer	0 = non required, 1 = required
3	action	Integer	0 = do nothing, 1 = compare, 2 = update only, 3 = update and reboot, 4 = readFile only
4	fileName	String	The name of the file to be updated
5	fileLocation	String	Path to the file to be updated
6	checksum	Integer	Checksum of the file to be read for readFile command

C.1.16 Configuration Update/Read Status [9000071.00]

Field #	Field Name	Integer	Description
1	errorCode	String	0 = successful, x = error code
2	fileName	String	The name of the file to be updated
3	checksum	Integer	Checksum of the file to be read for readFile command

C.1.17 Data Definition[9000080.00]

Field #	Field Name	Data Type	Description
1	recordId	Double	The data type and version
2	date	timestamp	Date the definition was created
3	dataDefinition	String	JSON encoding of an object describing this data type (See Appendix C1)
4			
5			
6			

C.1.18 QueryReport [9000085.00]

Field #	Field Name	Data Type	Description
1	user	String	User ID/ Name
2	limit	Integer	Number of results limit
3	tenant	UUID	UUID assigned by developer
4	startDate	timestamp	Date/ Time for range start
5	endDate	timestamp	Date/ Time for range end
6	queryTerms	String	String representation of filter conditions
7	Results	Integer	Number of results returned

Appendix D Change Management Strategy

The definition of a data record could undergo the following changes:

- A field could be **added** to the record.
- A field could be **deleted** from the record.
- A field in the record could be **changed** to a different data type.

In order to determine that one record type replaces another, we will alter the record version number.

New fields will always be **appended** to the definition so as to minimize disruption to existing code. This will maintain backward compatibility.

D.1 Merge

The Broker will handle the translation of data from one version to another. If a subscription is for one version, and another version is published, the Broker will implement the merge strategy for that data type and supply the Subscriber with data compatible with its requested version. It is the responsibility of the Application Developer to supply conversion syntax whenever a new version of the data is implemented. The map will consist of an XML string that is a copy of the data type's normal string except that the data supplied will be the default values.

e.g.

Original Version

```
<data Definition>
	<type>50050200.00</type>
	<recordName>invoiceDetail</recordName>
	<fieldCount>3</fieldCount>
	<dataRecord>
		<field>
			<fieldNo>1</fieldNo>
			<name>id</name>
			<data>0</data>
		</field>
		<field>
			<fieldNo>2</fieldNo>
			<name>description</name>
			<data>””</data>
		</field>
		<field>
			<fieldNo>3</fieldNo>
			<name>units</name>
			<data>””</data>
		</field>
	</dataRecord>
</data Definition>
```

New Version

```
<data Definition>
      <type>50050200.01</type>
      <recordName>invoiceDetail</recordName>
      <fieldCount>4</fieldCount>
      <dataRecord>
            <field>
                  <fieldNo>1</fieldNo>
                  <name>id</name>
                  <data>0</data>
            </field>
            <field>
                  <fieldNo>2</fieldNo>
                  <name>description</name>
                  <data>""</data>
            </field>
            <field>
                  <fieldNo>3</fieldNo>
                  <name>units</name>
                  <data>""</data>
            </field>
            <field>
                  <fieldNo>4</fieldNo>
                  <name>quantity</name>
                  <data>0</data>
            </field>
      </dataRecord>
</data Definition>
```

Notes:

1. Java data type conversions are handled by the Subscriber since all data is received as ascii and is translated by the application into the appropriate data type.
2. When a new version omits a field, the original field numbering is retained with a gap for the missing data.