

Chain of Events Analysis

Introduction

Control systems for the control of manufacturing processes have become increasingly complex. The interactions within such systems are often far from obvious, and require a great deal of time to manually examine using the traditional documentation tools – which were developed many years ago. In fact, the time required to do a thorough evaluation of the interactions is more often than not prohibitive when confirming proper system design. It is left to the user to perform spot checks, live system tests or simulations to try to confirm proper system design. The sheer number of possible combinations involved in most systems makes this impossible to catch all combinations. However, by the use of a structured database-centric analysis program each point within the system may be analyzed to determine the other points it can influence, and vice versa – what points can influence it.

Definitions

A “point” is an entity as defined by the manufacturer of the automation system.

A “parameter” is a particular piece of information (dynamic or static) related to a point.

Examples would be:

04FIC100 might be an entity, referring to a flow controller as defined within a DCS. PV might be a parameter of 04FIC100 referring to the flow controller’s current real-time process value. It is often referred to using the point.parameter syntax, and so “04FIC100.PV” refers to the 04FIC100 point’s PV parameter.

A PLC system would be similarly defined, with the point referring to a register or bit within the PLC memory or I/O cards, and the parameter referring to the points current value or internal tag name, description, etc.

Previous Tools

The typical existing system relies upon basic references. For example, you may call up the details of a single point and see what it is tied to – from its perspective only. But you will not see the full chain of point.parameter links that tie together. In most systems, you will not even see the other points that link to this point because the linkage is defined in the other point, not this one.

A typical PLC cross reference will show where a coil is defined, and in which relay rungs it is used, but to find out all the items that it can affect requires a tedious manual iterative process of examining each rung and in turn its cross references.

A typical DCS system will have a point detail display where that particular point’s linkages (defined by that point) are shown. In some cases, there is no way to determine what linkages may exist that are defined within other points. Or, this first level of linkages defined in other points may be found by the use of an offline tool.

Chain of Events Analysis

Interaction Software

By extracting or dynamically linking to each and every point and its linkage information, an interaction table is constructed that can be programmatically searched to define all point interactions. With this system it is now practical to ask the questions:

- What does this point, or point.parameter, influence?
- What can influence this point, or point.parameter?

These are two very powerful tools when verifying system design, and are the subject of US Patent 6,529,781 and other patent pending. These tools allow the user to create a list of all inputs that can possibly influence the state of a piece of equipment (e.g. pump, valve, compressor, conveyor, milling machine, etc.) or an internal relay coil or register (e.g. the master shutdown initiator or shutdown reset coil).

These lists can be printed out for verification, or displayed dynamically on the computer screen for analysis during trouble-shooting specific problems. See figure 1 for example of a printed report on a specific point.parameter:

Points influenced by PIC 6302.PVHHFL
XV_6302M.I1
XV_6304M.I1
FIC_4563.MODE
FIC_4563.OP
FIC_4564.MODE
FIC_4564.OP

Figure 1 Effects - basic data

This report is useful, particularly for a knowledgeable control system engineer familiar with the entire control system, but its value can be improved by linking this information with additional data within the database – such as point descriptions. See figure 2 for an example of the resulting table – which is of more value to everyone.

Chain of Events Analysis

Points influenced by PIC_6302.PVHHFL REACTOR R114 SECTION 1
XV_6302M.I1 C141 RECYCLE COMPRESSOR
XV_6304M.I1 C142 RECYCLE COMPRESSOR
FIC_4563.MODE C141 ANTISURGE
FIC_4563.OP C141 ANTISURGE
FIC_4564.MODE C142 ANTISURGE
FIC_4564.OP C142 ANTISURGE

Figure 2 Effects - with Point Descriptions

The prior tables were generated with an eye to ultimate results, skipping over intermediate values within the chain (e.g. an internal logic block that may be combining other information before causing the end result). This internal interaction is most effectively shown in an interactive display due to the large amount of information that may be present. Figure 3 is an example of such a display.

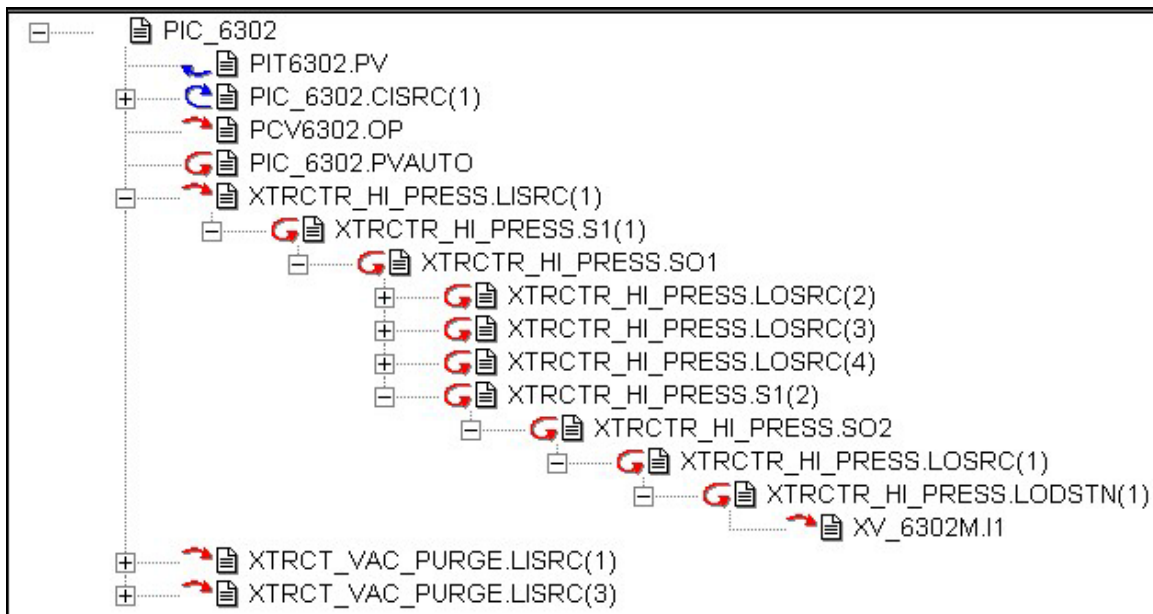


Figure 3 Interactions - Dynamic Display

Chain of Events Analysis

This information may also be asked from the reverse perspective, assuming you have an end result (say, the emergency shutdown master relay coil for a process unit). The user can ask the question “what can influence the state of this coil?”. The software can extract a list of all primary devices and setting that can influence this point.parameter. A resulting table may look as shown in figure 4.

Point.Parameter sources of influence upon 04ESD002 . PVFL
04PIC100 . PVHHTP
04PIC100 . PV
04XSD100 . DLYTIME (12)
04XSD100 . DEADBAND (4)
04FIC104 . PVLLTP
04FIC104 . PV
04XSD100 . DLYTIME (10)
04XSD100 . DEADBAND (4)

Figure 4 Influencers - basic data

These lists give the user a very well defined list from which to work for HazOps, operating manuals, troubleshooting, and confirmation of system interaction structure. The specific settings can also be extracted from the control system, adding another level of utility to the reports and displays. See figure 5 for an example of both the point description and parameter value data.

Chain of Events Analysis

Point.Parameter sources of influence upon 04ESD002 . PVFL UNIT 1 SHUTDOWN INIT		
Point.Parameter sources	Value	Units
04PIC100 . PVHHTP REACTOR TOP PRESS	245.00	PSIG
04PIC100.PV REACTOR TOP PRESS		
04XSD100 . DLYTIME (12) UNIT 1 SD LOGIC	30	Seconds
04XSD100 . DEADBAND (4) UNIT 1 SD LOGIC	2.5	PSIG
04FIC104 . PVLLTP REACTOR COOLANT	25.35	GPM
04FIC104 . PV REACTOR COOLANT		
04XSD100 . DLYTIME (10) UNIT 1 SD LOGIC	10	Seconds
04XSD100 . DEADBAND (4) UNIT 1 SD LOGIC	1.0	Percent

Figure 5 Influencers – with values & descriptions

Example Applications

This data is useful not only for initial system testing & commissioning, but also for verification of systems which have been in place for many years to detect latent faults. And, it is useful as a confirmation step before implementation of changes.

An example of where this would have been very useful is a modification that was made to a major process plant safety system. The system is extremely secure, using a TMR (Triple Modular Redundant) central logic system, triplicated analog field transmitters, and testable solenoid valves as final control elements. A new shutdown initiator was being added to bring the unit safely and quickly down upon total power failure to the process unit. The sensors were dutifully selected, carefully installed and wired, and the logic change was made. The logic was reviewed and appeared to be correct. After all, this was a relatively simple change just adding a new shutdown initiator to a proven system, which had been in operation for years. Unfortunately, the tag name of the new shutdown initiator was only one character different from the lamp test pushbutton.

A week or so after commissioning was complete, a routine test of the system ended with the required pressing of the lamp test pushbutton to confirm the operation of the indicator lamps – at which time the unit shutdown, bringing down the entire plant and an adjacent

Chain of Events Analysis

plant which also connected to this process unit. Upon review, it was found that a simple single keystroke error in entry of the tag name in one rung of logic was the root cause of this costly shutdown. In this case the result was an unplanned shutdown. It could just have easily been a failure to shutdown when needed, resulting in an even more hazardous situation.

The data can easily be expanded to include linkages to other objects – such as graphic displays & other data collection systems. This extensive reference is invaluable to maintenance personnel who must at times work on a particular instrument. They can now obtain a comprehensive answer to their number one question – “what might be affected when I disconnect and adjust this instrument?” It is known that the lack of an answer to such a question has caused numerous costly false shutdowns, and it is likely that this has also been the root cause of some disastrous incidents.

Implementation & Limitations

This system can work from a snapshot of the data, removed from the system and remotely analyzed, or from the live system itself. The live data extraction would not be totally in real-time, as the interactions would still require periodic scanning and analysis to keep current. Depending upon the system, different ways to keep the data accurate can be developed.

The offline approach has the limitation of being potentially out of date, but depending upon the procedural controls in place this may inject an acceptable amount of risk. The decision is highly dependent upon the user’s own organization and the criticality of his manufacturing process.

The database itself is the heart of the system, but to make the data useful an iterative interrogation of the data is required, with intelligent inclusion / exclusion of information to make the data useful to the user. For example, the software must be able to recognize what items are dynamically determined during operation (such as the process reading) and which items are configuration settings determined offline by the user. The handling of these two types of data is different.

Also, the software must have knowledge of the internal logic of the target control system, to understand how the data interacts given a particular configuration.

The system can determine the interactions internal to the control system (DCS, PLC, TMR safety system, etc.) but does not have knowledge of the process linkages. For example, the software can determine that a high pressure on the reactor (PIC_6302.PVHHFL) is linked to stopping a pump via a controller in the DCS (PMP_6302.I1) but it has no knowledge that this is turn will cause a low flow (FIC_6304.PVLLFL) which then in turn causes other actions to take place. This type of information can be manually added to the system, based upon process knowledge, but is imperfect in that it relies upon manual data entry.

Chain of Events Analysis

Summary

The programmatic analysis of control & shutdown logic systems can aid in preventing costly and/or dangerous incidents. It is now practical to answer basic questions about the design – such as “what can influence the state of this point?” and “what can this point influence?”

These answers are also of value to the routine maintenance and operation of a facility.

This interaction analysis software provides the user with one more tool to compliment others (such as on-line testing, simulation, logic reviews and HazOps) in confirming proper system operation, both for new systems, legacy systems, and ongoing modifications.