A Process Control Retrospective; What We Should Have Done

Edgar H. Bristol
Control Concepts Originated
28 Union St.
Foxboro, MA 02035
USA
Phone: (508) 543-8829
ebristol@mediaone.net

8/6/02
Fellow Emeritus
The Foxboro Co.

KEYWORDS

Process Control, Languages, Integration, Intent

ABSTRACT

Mid century process control was based on vendor companies usually competing on the basis of specialized contributions to the overall problem, each providing free application support to encourage the sales of their basic product. This encouraged design of cheaply applied products within the user's ability to cope, and represented a better incentive structure to the advance of the field. The current situation, where application effort represents a separate charge, either from the basic controls vendor or a separate supplier, encourages expensive clumsily designed tools, where the end user has insufficient resources to provide serious continuing support. The academic directions have fallen naturally in tune with this kind of practice, with advance techniques naturally focused toward direct end application rather than on tools that would be more easily used, tinker toy fashion, by application engineers solving some larger problem. And standards efforts have followed the same weakness in fundamental technology. A better arrangement would be to separate both teaching and practice of tool builders from their users so that sophisticated techniques are embedded in easier to use generalized tools. Application engineers and end users, who worked on a broader plane, could then use these tools, with controls then contributing in well-understood ways to the end solution. It is time to expect the digital system and control technology to contribute to a much more easily supported practice.

INTRODUCTION

Prior to the modern digital and electronics era (1955-60), process control related companies each depended on their unique control instrument technology which might be specialized about: sensor design, pneumatic/mechanical instruments, electrical instruments, etc. Field engineers often carried out application engineering as a free sales support service. The simplicity of the basic instruments made this feasible. Early digital efforts, on the other hand, depended on the general digital technology, generalized software expertise, with little application distinction. The absence of a uniquely valuable technology led to similar designs from all vendors, which lumped basic control software packages with specialized application software (leading eventually to strong standardization efforts). Whether the software was written in machine language, or, later, languages like C, the results for both standard control function and special applications were, and continue to be, much more difficult to apply. The broader range of application activities exacerbated this problem.

At this time vendors naturally advocated separate application charges. Of course, in actuality these charges were already built hidden into the overall system price. At the same time there were the internal company transitions as older engineers were pushed into facing the real dimensions of the new system application efforts. The author once argued with an older engineer who felt that the main role of a

particular (now generally eminently recognized) application engineer was to specify the instruments for a given sale. Even though this older perspective grossly misjudged the nature of the modern process control system, this paper will now mainly argue that the old way in fact contained the seeds of a better process control commercialism.

Historically process people had a great difficulty in understanding process control developments. It was a long time before university chemical engineering faculties developed the necessary control teaching. A part of this process was an evolution to more mathematical approaches to university teaching at odds with an also evolving application practice[1] and associated industrial teaching. In the early days of this evolution, the many industrial practitioners were able to defend their practice and there were few competent efforts at applying the academic ideas. More recent business practice has minimized available user control expertise and encouraged the use of outside control expertise on a one-shot design basis, with little continued support. And formal business justification practices, like their academic counterpart, limit acceptance of the more intuitive practice.

The results are detrimental to industrial control progress. The earlier free application of self contained, easily used products, generally maintained by the user, defines a better balance of incentives to vendors to evolve usable control technologies. On the other hand, separately charged application efforts (combined with standards forced commodity control software/hardware) encourage the vendors of those efforts to justify expensive design and separate (and separately charged) design documentation.

Before this current era, interest in more complex control systems was encouraging industrial practitioners (most with analog[1,2] rather than digital experience)[2] to evolve a practice with the following desirable characteristics:

- Use of standard modules such as PID controllers or lead lag dynamic compensators, with standard operators for square root, addition, multiplication, etc.

[The conventional digital approach maps this into a software counterpart. A newer, more powerfully digital, proposal,[3] by the author, distinguishes between broad application strategies (Idioms), simpler standard compensation functions as above, and normal assignment statement computations.]

- Required only simple algebra to represent the underlying process characteristics (for instance in a feed forward or decoupling system) or simple LaPlace transform dynamics. In the chemical engineering environment, with its basic conservation laws, this kind of algebra allowed transparent accommodation of basic underlying process nonlinearity.
- Standardized strategies for accommodating those issues not accommodated by the simple algebra derived structure by tuned feedbacks and dynamics.
- When applied to a given process, the resulting application lent itself to teaching of the working production engineers, making them participants in keeping that system useful and functional.

The experts in this practice had a deep understanding of the process, which they easily related to control designs, without dependence on the more refined control mathematics. They lacked both the mathematical expertise **and its need**. One can compare what happened when the mathematically sophisticated control graduate came in contact with these engineers to what might happen if an experienced plumber was brought in contact with a practice inexperienced hydrodynamicist to discuss

---

[1] Admittedly the practice had its own problems: presumably well designed or tuned control systems that were soon put in MANUAL. But its replacement by more sophisticated mathematical methods was not a fix for such problems. What is always needed is available on-site talent and discipline to keep the system functional, or a technology that is either (as is unlikely) perfect, or designed to be within the capacity of the on-site engineers.

[2] The paper addresses the continuous control practice, not the equally important practice of logical and sequencing (batch) control which was also separately evolving.

plumbing.

The problem is not basically one of the practitioner or the mathematician being right or wrong. (But indeed, under current business attitudes, we are in great danger of losing the value of the practitioners.) The problem is that we have abandoned the old distinction between the specialized role of tool designers using refined and sophisticated techniques (including advanced mathematics) and application experts with the greater breadth and just that detailed control understanding needed to effectively use those tools. Necessarily in the modern era, the specialized designer will need to go beyond the level of mathematical or software understanding of the user. But he will have to learn enough of the application issues to best tailor the hidden sophistication in the interests of the user. Admittedly this is a difficult task, certainly not satisfied by watering down the flexibility and optimality of the end result.

In the vendor organization, the loss of the distinction has placed the designers with broad application experience in a stronger position but has meant that there are no longer engineers developing specialized proprietary capabilities based on software (or mathematics) which can lead the field in ease of use. The result is application engineers working in the more difficult areas of software, without the necessary dedication to quality software and design documentation (whose cost, in a proper design, would be positioned to be amortized over many end sales). At the same time no one is focusing on the novel designs, which would go beyond standard graphics/GUI usage to achieve real ease of use. The special talent for the quick and dirty design of one of a kind applications which should be the application engineers contribution finds no role because, catch 22, the available tools are too difficult to allow such a practice to prevail.

One cannot imagine how the existing process control business would reconstitute itself to permit the evolution of the expert controller designers, even at the time when software has begun to approach a level where ease of use is possible. Nor is it easy to see how the user community would reestablish its own application control expertise, married to the end user's desires and needs. But both are needed.

The general nature of systems and computer programming technologies both limit the needed advances. Both technologies take am ability to implement any solution as central to their character. For this reason invention (and patenting) are foreign to both forms of practitioners. A clue to resolving this thinking can be taken from an observation of another older engineer: that systems "was a bag of tricks". Innovation then calls for creating inventing a new trick (like the PID) or inventing a new bag (e.g. improved control framework).

The PID is usually taken as the central symbol of traditional process control. While this is by no means the only tool of traditional practice, the paper will illustrate the broader issues with a discussion of how the PID is still the potential beneficiary of specialized treatment, both in a mathematical and a software dimension.

HOW TO BETTER UNDERSTAND AND ADVANCE THE PID

Because of the broad character of feedback, the PID is the most familiar tuned function in process control. While there are many direct or quasi-algebraic methods for tuning the PID, in fact such tuning rules are almost never as general and effective as experienced manual iterative tuning.[3] As for mathematical methods, even though manual tuning is no more difficult to learn than, for instance, learning how to stop a car efficiently for a traffic light, there is no accepted, effective, formal understanding of PID tuning. The most effective adaptive or automated tuning PID systems[4,5], automate experience, with

[3] Because of the inherent impossibility of precise process modeling, computed tuning methods can only result in unpredictable or deliberately loose tuning. In this case, the best manual tuning must depend on iterative tuning driving the actual process response to the desired response shape.

only loose connection to theory. This section will discuss the problems of such a theory.

Given that the essence of a PID is its role as a manually or automatically tuned device, a complete theory is one that explains, well enough for **exact** prediction, the relation between the parameters and the expected controlled response. Any PID controversies are a consequence of the inexactnesses of the arguments we use in place of such an exact theory. The problem includes a number of dimensions:

1. Normal on line measurements are never adequate to identify more than a very low order model.
2. A good computed closed loop model requires accurate modeling of the higher terms in the corresponding open loop model.
3. Normally used input output modeling tends to generate a best-fit open loop model (and therefore by items 1 and 2, a poor closed loop model) even under closed loop conditions. One reference[4,6] shows an example where a nearly unstable (cycling) closed loop response is nevertheless identified, under normal low order open loop modeling, as a process with highly stable computed closed loop.
4. Most fundamentally, under normal fully algebraic methods, the time (tuned) response behavior is not analytically computable as a function of the tuned parameters; the parameters are certainly not accessible analytically from the response.

No process control theory based on an assumed exact model can ever be honest; such a model is not remotely possible. Thus a good theory of the PID is always about approximation. Such a theory will always be more difficult than an exact theory. The author[7] argues that, to very high accuracy, the closed loop response is computable as the sum of the four dominant eigenfunctions[4], filtered by the input to output disturbance transfer function. That reference begins to develop a theory where Taylor series methods can be used to compute general algebraic expressions for the desired closed loop eigenfunctions from an open loop model. This addresses the underlying Problem 4 above.

Such a theory can be used to analyze and improve the existing performance feedback (pattern recognition) adaptive control methods[4] or design a more directly mathematical technique using a priori assertions about the unattainable higher order open loop behavior. Alternatively, one can get greater identification data if the identification includes disturbance responses measured under distinctly different sets of parameter values.[5] These PID issues represent just one area where a more serious tool oriented approach still has much to contribute.

Beyond the PID theory there are many other algorithmic and software issues lending themselves to cleaner treatment (in the context of the Idioms below). The author has published useful control strategies, naturally a consequence of traditional practice (e.g. higher level forms of split range control), which are far too clumsy to implement in the usual block framework. Conventional computer programming structures and implementations (e.g. with DO/FOR loops sharing a common set of control parameters) can make such strategies practical and even elegant.[8,9]

ADVANCED CONTROL

The distinction between academic and practical control is most apparent in more advanced control. Academic advanced control has tended to focus on complex linear regulatory and constraint oriented multivariable control. In practice, the more interesting dimension has been simple techniques addressing widely different kinds of coordinated controls and constraints arising from the different application economic and safety requirements (e.g. fuel air combustion control and blend pacing).[1,2,10] Time is

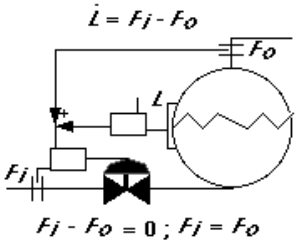
[4] Tight derivative control or deadtime processes may require one or more additional eigenfunctions to capture the full response shape (their absence does not adversely effect tuning decisions).

[5] This is one of the reasons why iterative tuning is essential: it provides the different parameter values and identifications supporting the implicit high order open loop terms.

inadequate to cover these inventive approaches; discussion will be restricted to the treatment of conventional multivariable control.

For the practical analog control tradition the natural advanced control representation has been the block diagram without any linear restriction. For the linear control this has emphasized feedforward and decoupling. As general tunable techniques (like the PID) these have lent themselves to processes like paper for which the basic modeling equations are not well understood. In this case, both static and dynamic decoupled dimensions may be tuned, manually or automatically.

A more elegant form of control practice derives the feedforwards and decouplers from the chemical engineering balance laws of the form: $\partial x/\partial t = f(x, d, u)$. The form of the feedforward or (partial or full) decoupler is given by setting the derivative to zero: $f(x, d, u) = 0$ (which corresponds to making the controlled variable constant). The earliest example is three-element boiler control.



In this case, the level in a tank of boiling water (whose outlet steam is a free variable) is to be controlled by manipulating the inlet (cold water) flow. The difficulty is that increases in the cold water inlet tend to cause the water level to settle in the short term rather than increase. The equation for the level **L** and the resulting feedfoward control is shown in the figure. As typical, feedback is included in two forms: as the secondary flow control for the inlet flow $\mathbf{F_i}$ and as the primary level control to trim the inherent feedforward inaccuracies.

The parallel of this traditional analytically nonlinear feedforward design to academic state space control is not usually recognized, yet such methods are appropriate in establishing basic compensating structures when not called on to define actual feedback controls. When applied to feedback control, state space control techniques can be unpredictably sensitive; the use of tuned controllers is more predictable. In practice, no such feedforward design can be accurate enough without feedback, which is most naturally structured as the source of certain uncertain parameters in the feedforward calculation. Similarly it is often desirable to apply (tuned) dynamic compensators in such a model. Practical adaptation of feedforward and multivariable dynamics is now available. The Moment Projection method[4,11] particularly lends itself to combination with traditional methods

Multivariable control transcending the PID practice under the variations of model predictive control should be mentioned. In themselves they represent a direction contrary to the argued separation between sophisticated tool building and simple application. They also complicate the direct use of process nonlinear behavior. The commercial acceptance of such methods in refinery practice is, in this respect, justified by the fact that such processes tend to be operated at a single (and therefore linearizable) full production operating point. But their overall complexity may also preclude a more fluid approach to control, and further compound the problems when changes must be made. Unpublished work is considering how better to achieve greater fluidity and integration.

IDIOM CONTROL NOTATION AND LANGUAGE

One of the reasons, that the mathematical methods have held sway in the university work is that there has been no academically useful way of teaching students about the larger (above PID) dimensions of the practical control application.[6] The author's Idiom work[12,13] originated as an approach to being able to relate the basic control teaching into a higher level perspective and better document the way the control loops were actually used. One thought is that the first couple of days in a semester course might be taught discussing control of processes that the students might have encountered, captured in the Idiom notation.

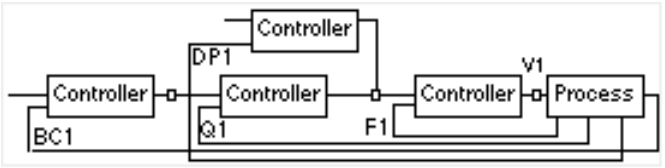
[6] In that classroom only theory can be taught, and even in the academic laboratory, random observations unrelated to a theory are soon lost.

This would then serve as a larger background as the students descended into the study of the single loop.

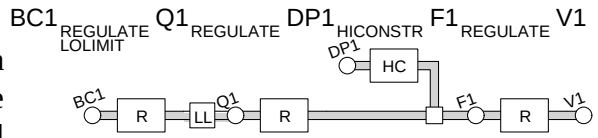
Since that initial work the notation has evolved into a general-purpose control language proposal[10,14-16] that includes not just the PID based traditional loops but multivariable controls, logic, batch sequencing. The language focuses on integrating controls, as practiced, and supporting a readable self documentation never seen in the industry, made possible by a higher level notation emphasizing the control intent rather than its implementation. This same higher level allows the automatic integration and configuration of control related functions such as the human interface and, more, recently process system maintenance.[17]

As part of the referenced website, in addition to published papers, other demos, and user's guides, the author has implemented a documentation tool based on the language Idiom Notation, with the hope that some teaching using that tool might sometime take place (or even serious documentation of real applications!). The author is working to develop some university experiences hopefully soon to be published. The general character of the basic notation is outlined here.

The figure shows a triple stage cascade taken from a well-publicized debutanizer example.[10,12,15.16] The diagram corresponds to a single Process Control control loop.



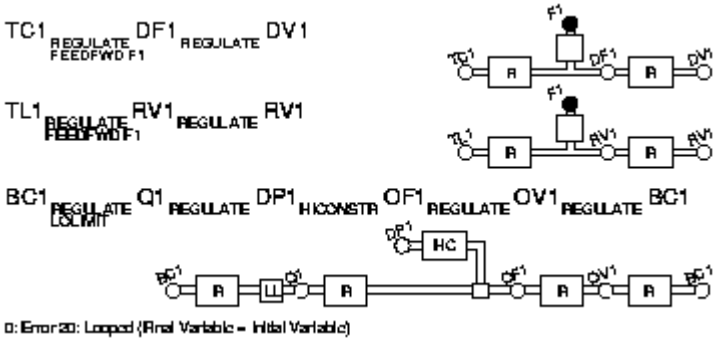
[In process control terms this is a single loop because the nested loops are supporting parts of the primary loop. The basic goal of the control system is control of the Bottoms Composition (BC1) by manipulating the Valve (V1). The subordinate loops act to improve the resulting control.]



Each Idiom Loop Statement in a multiloop design corresponds to one primary controlled variable and one final manipulated variable. The secondary cascaded controlled variables in the statement (to the right) represent successive variables along a Degree of Freedom Path through which the primary composition variable is controlled ultimately leading to the valve. As each measurement is manipulated to control the earlier stage, it is itself controlled by being subject to regulation and other limitations (e.g. Hi or Lo Limits, Hi or Lo Constraints) and refinements (e.g. Feedforwards). In this way, each measurement in the Degree of Freedom Path is associated with a Phrase (e.g. $BC1_{REGULATE\ LOLIMIT}$) repeating a basic language statement regulatory pattern.

Starting with the primary variable (BC1 in the statement), until the final actuator Phrase, the successive Phrases each consist of a controlled variable and a simple or compound subscripted Idiom operator usually initiated with a REGULATE Idiom representing the intent to regulate the variable. The BC1 Phrase also includes the LoLimit Idiom representing the intent to low-limit the output target to the following heat flow variable Q1. The HICONSTR Idiom operator associated with the differential pressure variable DP1 represents a high constraint on DP1. It represents a "higher precedence operator" (like multiply over add) indicating that the constraint PID will take over from the heat flow controller if the constraint is violated. Whichever controller prevails drives the Flow variable F1, whose control than drives the valve V1. The details of the Idiom Loop Statement, including its MIMO generalization, are developed in more detail in the references.[10]

The traditional block diagram is focused on the functions and connections implementing controls, less meaningful in a computer where a compiler manages all connection, and certainly less clear to someone concerned with the intent of the controls or with a student trying to get the sense of a large system. In addition it imposes only local



restrictions to the connections between blocks. The tool illustrates the far greater power provided by the Idiom Loop Degree of Freedom Intent model. Such a model expresses global constraints, not merely local constraints. For example, the block diagram would permit the connection of the V1 back to the setpoint of the BC1 in a Degree of Freedom loop, as in the bottom statement in the adjacent figure. Nothing about the individual rules of controller connections prevents that, even though it is an absurdity from a control point of view.

The documentation tool compiler error checks the input, based on the Degree of Freedom Path model. This is as essential to such a tool as to any compiler and as a spelling checker is to a word processor. The value of the spelling checker goes far beyond checking spelling errors per se; it recognizes mistypings; it "proof reads". The deeper the basic Intent "understanding" built into the language system, the greater the clarity, ease of use, and integration provided. In addition to the looping error, the tool checks for any other Degree of Freedom violations, involving any statement or collections of statements. Translated into control language design these kinds of error checking become essential and normal. In the same way they define (for possible automatic implementation) the appropriate integrated display actions for a corresponding operator display. The referenced maintenance discussion[17] depends on a Predictive Maintenance function to predict device failures, an explicit cost model to express the economic consequences of failure to control any variable, and the Idiom Loop Statements to relate controlled variables to failures affecting them. The result can be a maintenance scheduling display that projects the projected costs of all continuing failures. This rationalizes the maintenance response.

CONCLUSIONS

Process Control is in a quandary. The digital systems, which should make life easier, continue to make it harder. And at a time when digital expertise is beginning to face up to the consequences of digital complexity. On one hand we no longer have vendors of simple aids to control. Instead they apply their resources to interfacing into their systems the latest general communications, graphics, or computational capability. Never mind a determination of good process control uses of these capabilities. And yet these capabilities could present unique value, if appropriate experts developed them. But, lacking the inventiveness and discouraged by users and standards, the vendors no longer look to this kind of unique (yes, proprietary) contribution. And so, instead of round robin examination of their process for easily implemented improvements, application engineers must sweat over obscure digital technology not covered in the Units Op course.

A restoration of the traditional user/vendor division of labor would act to restore a real progress to the field. This would be helped by an academic sensitivity to the difference in tool building and application roles. Normally, academic work of a more practical nature is impossible to achieve; when university pedagogy of complex issues abandons theory it tends to become unteachable. But it might just be that a different kind of theory might be both teachable and more appropriate to the development of a powerful practice. Among other things, the paper has discussed a more computer compiler technology oriented kind of theory whose benefit is the appropriate organization of large engineered control systems. Such an approach then becomes suggestive of all kinds of associated low and high level tool and application problems that might just accidentally map better into the needs of practice.

REFERENCES

The author papers below are available on the web site: http://homepage.mac.com/ebristol, along with related demos and User's Guides.

[1]F.G. Shinskey, Multivariable Control, ISA, '82.

[2]F.G. Shinskey, Process Control Systems, McGraw Hill Book Company, '88.

[3]E.H. Bristol, "A Field Device Language in a Process Control Language Family", ISA2000, New Orleans, Aug. ‘00.

[4]E.H. Bristol, "Process Systems Adaptation Beyond Math Modeling", Workshop on Adaptive Control Strategies, Lake Kananakis, Alberta (organized by the Dept. of Chemical Engineering, the University of Alberta, June 20-22, ‘88, pp. 336-354.

[5]K.J Astrom and T. Hagglund, PID Controllers, Theory, Design, Tuning, 2nd Edition, ISA, Research Triangle Park, NC, ‘95.

[6]E.H. Bristol, “Experimental Analysis for Engineering of Adaptive Designs”, Chemical Engineering Progress, June 1983.

[7]E.H. Bristol, "Understanding the PID and its Tuning", PID00, Terrassa, Spain, April 20-24, ‘00.

[8]E.H. Bristol, "Basic Control Algorithms", Process Instruments and Controls Handbook, (3rd and 4th Edition, McGraw Hill Book Company, New York (‘85 and) ’93, pp. 17.158-17.175.

[9]E.H. Bristol, "Even the PID Needs Theory/Software Advances", 1994 Control Engineering Exposition, 3rd International Control Engineering Conference, McCormack Place, Chicago, Mar. 14-16, ‘94.

[10]E.H.Bristol, "Idiom Based Documentation of Continuous (Feedback) Process Control", Documentation Tool User’s Guide, http://homepage.mac.com/ebristol website, most recent version: Apr. 18, ‘01.

[11]E.H. Bristol and P.D. Hansen, "Moment Projection Feedforward Control Adaptation", 1987 ACC, Minneapolis, MN, June 10-12, ’87, Session FA10.

[12]E.H. Bristol, "Strategic Design: A Practical Chapter in a Textbook on Control", 1980 JACC, San Francisco, CA, Aug., ‘80.

[13]T.J. Prassinos, T.J. McAvoy, and E.H. Bristol, "A Method for the Analysis of Complex Control Schemes", 1982 ACC, Arlington VA, June, ‘82.

[14]E.H. Bristol, "A Language for Integrated Process Control Application", Retirement Symposium in Honor of Prof. Ted. Williams, Purdue University, West Lafayette, IN, Dec. 5 - 6, `94.

[15]E.H. Bristol, "Not a Batch Language; A Control Language", World Batch Forum, San Francisco, May `95; also ISA Transactions, Dec. `95.

[16]E.H. Bristol, "Redesigned State Logic for an Easier to Use Control Language", World Batch Forum, Toronto, May 13-15, `96.

[17]E.H. Bristol, "Configurators, Languages, and Graphics", ISA2001, Houston, Sept. 10-12, ‘01.