How to lose money with basic controls

Follow these 21 rules to ensure that many opportunities to minimize process disturbances will be completely overlooked

M. J. KING, Whitehouse Consulting, Southampton, UK

An earlier article introduced the "Whitehouse rules" which, if followed, would ensure almost total failure of an advanced process control (APC) project.[1] The article's intent was, of course, to help sites avoid the common pitfalls. It failed completely to do so.

Indeed, since then, the industry has discovered a whole new range of ways of getting things wrong. So many, that a single article would not do them all justice. Instead we focus here on basic controls. Rest assured that there remains plenty of material for future articles on other aspects of APC implementation.

1. Don't bother training control engineers in basic control techniques. This rule is first for good reason. Adhering to it will result in just about all the others also being followed. The industry does not question sending engineers on courses dealing with advanced control products but seems to assume that they will absorb the requisite basic control knowledge over time. They may eventually, of course, but not until the harm is done. See Rule 2.

2. Progress APC implementation before finalizing the basic controls. Remember that changes to the basic controls will affect overall process dynamics. Unless completed prior to step testing for multivariable control design, the site has effectively committed to retaining poor basic controls until the next process revamp. Step testing is usually a major exercise that no one will want to repeat until process changes require it.

3. Rely on the APC vendor to sort out the basic controls. The APC vendor will of course highlight, usually during pretesting, any controllers that appear not to be working well. However, the APC vendors are just as guilty as the sites in not training their personnel in basic control. The vendor's engineers may not even appreciate what improvements are possible. Their prime concern is to ensure the controls are working sufficiently well to support multivariable control. This is a far less demanding criterion.

4. Award the APC implementation contract before resolving the basic controls. The vendor will be under its own budget and schedule pressures. It will not wish to become involved in time-consuming changes to the basic controls unless not doing so will seriously impair multivariable controller performance. The site will feel under pressure to let the vendor begin step testing before all the basic controls have been properly addressed.

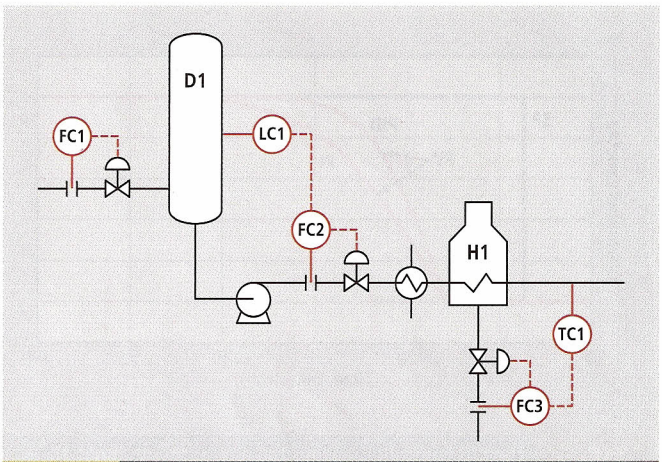

FIG. 1. Different PID algorithms affect control of this process.

5. Apply rigorous design checks to all controllers before awarding the APC contract. While in theory, performance of all controllers can be improved by good design, for many there is little point. While it is common to be able to at least halve the time that it takes a controller to respond to process disturbances, little would be gained, for instance, by improving a flow controller that already responds within a few seconds. After all, it is the APC—rather than the basic controls—that should capture most of the benefits. Unnecessarily delaying its commissioning does not make economic sense. The aim is to identify those controllers where additional design effort is justified. In general, these would be those where the process dynamics are relatively slow and those where special attention is needed (See Rule 8).

6. Always apply the conventional version of the PID algorithm. Most DCSs offer a variety of PID algorithms. Unless the engineer has thoroughly read the user manual, he or she may not be aware of these. Even then, why there are so many and which should be used under what circumstances may not be appreciated. Not fully appreciating the benefits, either the default or the one that most closely matches the engineer's understanding of PID control will be chosen. This decision is almost always wrong. In making it, the opportunity to reduce—often by a factor of three—the duration of process disturbances will be missed.

This is illustrated by considering the process outlined in Fig. 1.

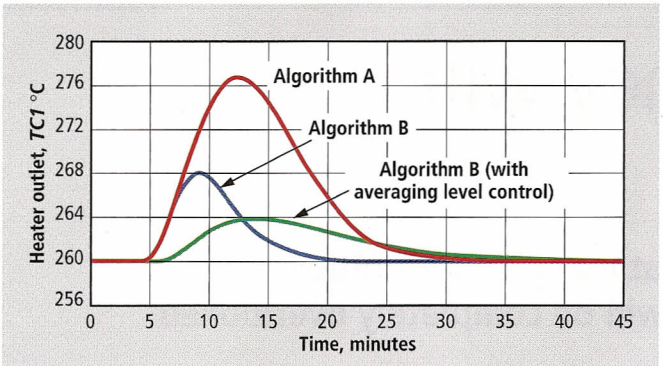

FIG. 2 The algorithms require different tuning constants.

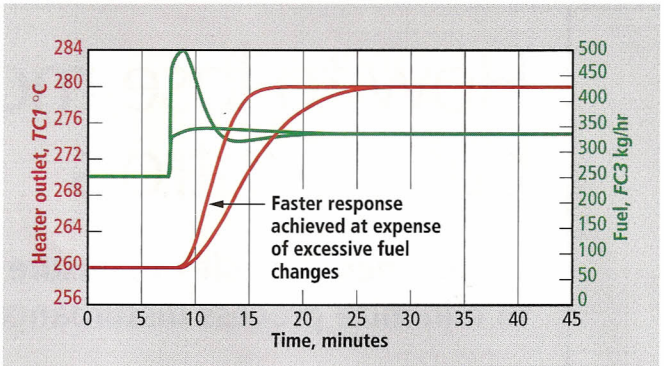

FIG. 3 Improvement in response time has been gained at the expense of rapid changes to the fuel.

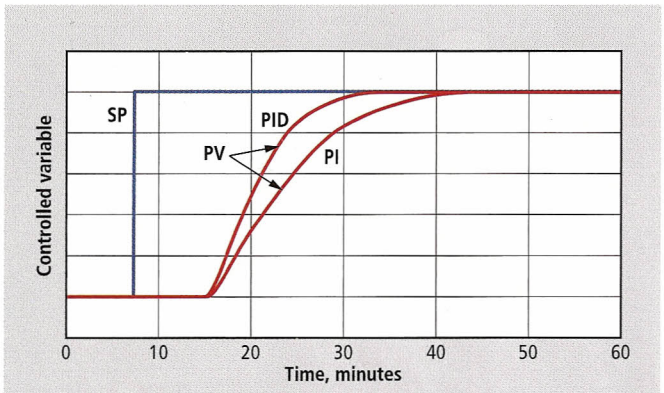

FIG. 4. The PID controller reaches the new setpoint about 10 minutes sooner.

Fig. 2 shows performance of the heater outlet temperature controller (TC1). Algorithm A is the conventional PID controller; algorithm B is also available in most DCSs. They require different tuning constants but were tuned optimally to give the same response to setpoint changes. They were then subjected to the same feed rate disturbance by reducing the FC1 setpoint.

Other versions of the PID algorithm have similar benefits. Some avoid the "derivative spike" that can result when setpoint changes are made. Others have nonlinear features, such as error squaring—particularly beneficial for level controllers.

7. Switch algorithms automatically as controllers are put into cascade. In principle, some versions of the PID algorithm are better suited to setpoint changes, while others better handle process disturbances. Standalone controllers are subject to relatively few setpoint changes when compared to the number of process disturbances. Secondaries of cascades, however, are subject to frequent setpoint changes. There is an argument, therefore, to use different PID algorithms for each of these circumstances.

This begs the question, however, as to what to do when the operator breaks a cascade. One DCS vendor at least offers the option of automatically switching algorithms. While at first sight a good idea, it neglects the fact that the two algorithms require very different tuning. Control degradation resulting from failing to change tuning constants far outweighs the small benefit of the algorithm change.

TABLE 1. PID algorithm types

Algorithm	Noninteractive	Interactive
Other names	"Parallel" "Ideal"	"Series"
Laplace form	$M = K_c\left[1+\frac{1}{T_i s}+T_d s\right]E$	$M = K_c\left[1+\frac{1}{T_i s}\right]\left[1+T_d s\right]E$

M = controller output
K_c = controller gain
T_i = integral time
T_d = derivative time
E = error

8. Always tune for fast return to setpoint. While it is true that the objective of most controllers is to return the controlled variable to the setpoint as soon as possible, slavishly following this for all controllers is a mistake. Many level controllers, for example, should be set up to make use of surge capacity. The controller should be designed to, following an upstream flow disturbance, make the minimum change to the manipulated variable (the downstream flow), keeping the level with alarm limits and then slowly returning it to setpoint. The benefit of this "averaging" level control is clear from Fig. 2.

However, manipulated variable movement is not only a consideration for level control. Consider setpoint changes made to the temperature controller in Fig. 1. Fig. 3 compares controller response tuned without considering fuel disturbances to one where less aggressive tuning has been applied. Improvement in response time has been gained at the expense of rapid changes to the fuel—so rapid that firing was temporarily increased to maximum. Almost certainly the combustion controls would not respond quickly enough, leaving unburned fuel in the flue gases.

9. Assume that proportional-only control is always impractical. Most engineers are aware that this type of controller exhibits offset, although few can explain why. It would appear that such a controller should not be considered. There are exceptions. For example, the integral time calculation for averaging level control (see Rule 8) can occasionally lead to a value that is greater than the maximum supported by the DCS. Restricting it to this value will underutilize vessel surge capacity. It may be better to accept offset on vessel level in return for less variation in downstream flow.

10. Only apply derivative action to temperature controllers. This is part of process control folklore. Derivative control is highly beneficial if the process deadtime, θ, is large compared to the process lag, τ. Its anticipatory nature helps the controller respond much more quickly to disturbances. Many temperature controllers have such dynamics, while most other controllers do not—hence the folklore.

However, many temperature controllers have negligible deadtime. Use of derivative action here will bring little benefit and can cause stability problems. Similarly, there will be many other controllers where θ/τ is much greater than unity; ignoring the benefit of derivative action will greatly extend time taken by the process takes to recover from disturbances.

Fig. 4 compares response of an optimally tuned PID controller with an optimally tuned PI controller. In this case, where the process deadtime is around eight minutes, the PID controller reaches the new setpoint around 10 minutes sooner.

11. Apply derivative action to controllers with discontinuous measurements. Following the argument presented in Rule 10, it would seem that derivative action should be applied to any controller that takes its measurement from an onstream analyzer. Such analyzers, through sample delays, can introduce large deadtimes.

However, many analyzers, such as chromatographs, produce discontinuous signals. Steps in the signal appear to the controller as very high rates of change and will cause large derivative "spikes." While there are PID algorithms that can deal with setpoint changes without causing such spikes (see Rule 6), none handle process changes in the same way. There is little choice but to use a PI controller or apply a special-purpose deadtime compensation algorithm such as Smith or IMC.

There is an increasing tendency to install digital transmitters. Care must be taken with these. Some have resolutions of, for example, 0.1% of instrument range and, therefore, even continuous signals will exhibit steps as they change. While a 0.1%-step may seem very small, it is enough to cause large derivative spikes and preclude use of derivative action.

12. Ignore the controller scan interval. While it is reasonable to assume that a digital controller with a scan interval of one or two seconds is a close approximation to analog control, this is only the case where the process dynamics are on the order of minutes. There are processes—for example, some compressor controls—that have much faster dynamics. Applying a tuning method based on analog control will result in the controller being too tightly tuned—possibly to the point of being unstable.

13. Ignore the difference between interactive and noninteractive PID control. Table 1 describes the two algorithms. Some DCSs offer only one. Others offer the choice. If no derivative action is included, the algorithms are identical. However, their tuning is quite different if derivative action is required (see Rule 11). Many tuning methods fail to indicate the algorithm for which they are designed.

14. Tune by trial-and-error. Certainly this method has its attractions. It requires very little knowledge about process dynamics or the control algorithm. It works with the real controller and real process with the controller in closed loop. Its main drawback is that it is incredibly time-consuming. An effective design tool will allow optimum tuning for a temperature controller, such as that in Fig. 1, to be completed in a couple of hours.

By trial-and-error, getting to the same point will take around 40 hours. No organization can realistically dedicate this amount of time to a single controller, particularly since there are likely to be several hundred others like it. In practice, tuning is stopped when the parameters are "about right," resulting in a controller poor at handling any significant process upset.

15. Tune using published methods. Almost every month, a journal publishes a new tuning method or a new product is released. With few exceptions, these are usually flawed in some way. Very few take account of controller scan interval (see Rule 12), although this is only a problem in special circumstances. More importantly, almost all ignore the manipulated variable (see Rule 8). The problem of excessive manipulated variable overshoot is likely as the process deadtime approaches zero or, more correctly, becomes small compared to the process lag. Such processes are very common.

This, together with situations where we wish deliberately to design for a slow return to setpoint, invalidates the tuning method for maybe half the site's controllers. Finally, the tuning method may not match the PID algorithm version being applied. Different DCS vendors implement even the traditional PID controller in different ways. Plus, within any DCS, there is a range of optional changes to the traditional approach (see Rule 6). Failure to take account of this may make the tuning method invalid for all of the site's controllers. The key is to carefully evaluate any method to ensure it takes account of all the above.

16. Install self-tuning controllers. These suffer many of the problems of the published tuning methods (see Rule 15). While those supplied by the DCS vendor should match the PID algorithms implemented in its system, they still often use incorrect tuning criteria. In practice, for a variety of other reasons, there are almost no successful installations.

They can also involve costly license fees or hardware modifications. Importantly, they are no substitute for a true analysis of the problem. From an understanding of the process dynamics and how they vary (or if indeed they do), it is usually possible to design much simpler modifications to the control scheme to handle any problems associated with tuning.

17. Forget that process gains vary with feed rate. Think about heater outlet temperature gain, *TC1*, with respect to fuel changes, *FC3*. Remember, gain is defined as the resulting change in temperature divided by the change in fuel flow, i.e., $\Delta TC1/\Delta FC3$. At low feed rates, a fuel change will produce a large temperature change. At high feed rates, the same fuel change will produce a much smaller temperature change. This effect is not unique to heaters but will apply to almost every controller.

Since process gains vary inversely with feed rate, to maintain constant loop gains, the controller gains must be adjusted in proportion to feed rate. Fortunately, most units do not experience feed rate changes large enough to cause tuning problems. A gain change of about 20% is usually required for control degradation to be noticeable. This would require a unit turndown of 1.5 (from 120% to 80%). Turndowns greater than this would justify some form of adaptive tuning.

18. Only implement feedforward control if there are frequent disturbances. While feedforward control aims to "warn" the controller to take corrective action before it experiences the disturbance directly, it does have a secondary benefit. Use of ratio feedforward control resolves the problem caused by process gains varying with feed rate. Configuring the temperature controller (TC1) to manipulate a fuel-to-feed ratio will remove need for the controller gain to be changed whenever feed rate changes. Obviating need to retune the controller would be beneficial even if feed rate changes occur only every few weeks.

19. Always filter noisy measurements. There is a tendency in the industry to filter noisy measurements simply based on making them look better on trends. The problem is that filters also change the process dynamics—usually for the worse. This means that the controller tuning has to be relaxed, making it less able to deal quickly with process disturbances. The important criterion is what noise level is passed through to the final actuator, where excessive noise can indeed do real harm. This depends not only on the noise amplitude but also on controller tuning. Derivative actions and high controller gains amplify noise and may, therefore, justify a filter. PI controllers with low gains usually do not.

20. Install the filter after the controller is commissioned. As stated in Rule 19, filters change the dynamics. Implementing one after a controller has been tuned will reduce controller stability—possibly to the point of instability.

21. Always use the standard DCS filter. This is usually a first-order exponential filter, i.e., a lag. While a good general-purpose filter, it increases both deadtime and the lag "seen" by the controller. While the controller can, of course, be tuned for the revised dynamics, its performance will certainly degrade. Whether this is noticeable depends on the size of the filter lag compared to the process lag. Other filters, such as the least-squares filter, can provide comparable noise reduction without having such an adverse effect on dynamics. While they will usually need to be especially coded in the DCS, this is relatively simple to do and can readily be cloned if required for many measurements. **HP**

LITERATURE CITED

[1] King, M. J., "How to lose money with advanced controls," *Hydrocarbon Processing*, June 1992.

Myke King founded Whitehouse Consulting in 1992. Previously, he was a founder member of KBC Process Automation and prior to that was employed by Exxon. Mr. King is responsible for consultancy services assisting clients with developing and executing advanced control and information system projects. He has almost 30 years of experience in such projects, working with many of the world's leading oil and petrochemical companies. Over 1,000 engineers have attended his training courses. Mr. King has an MA in chemical engineering from Cambridge University and is a Fellow of the I Chem E.