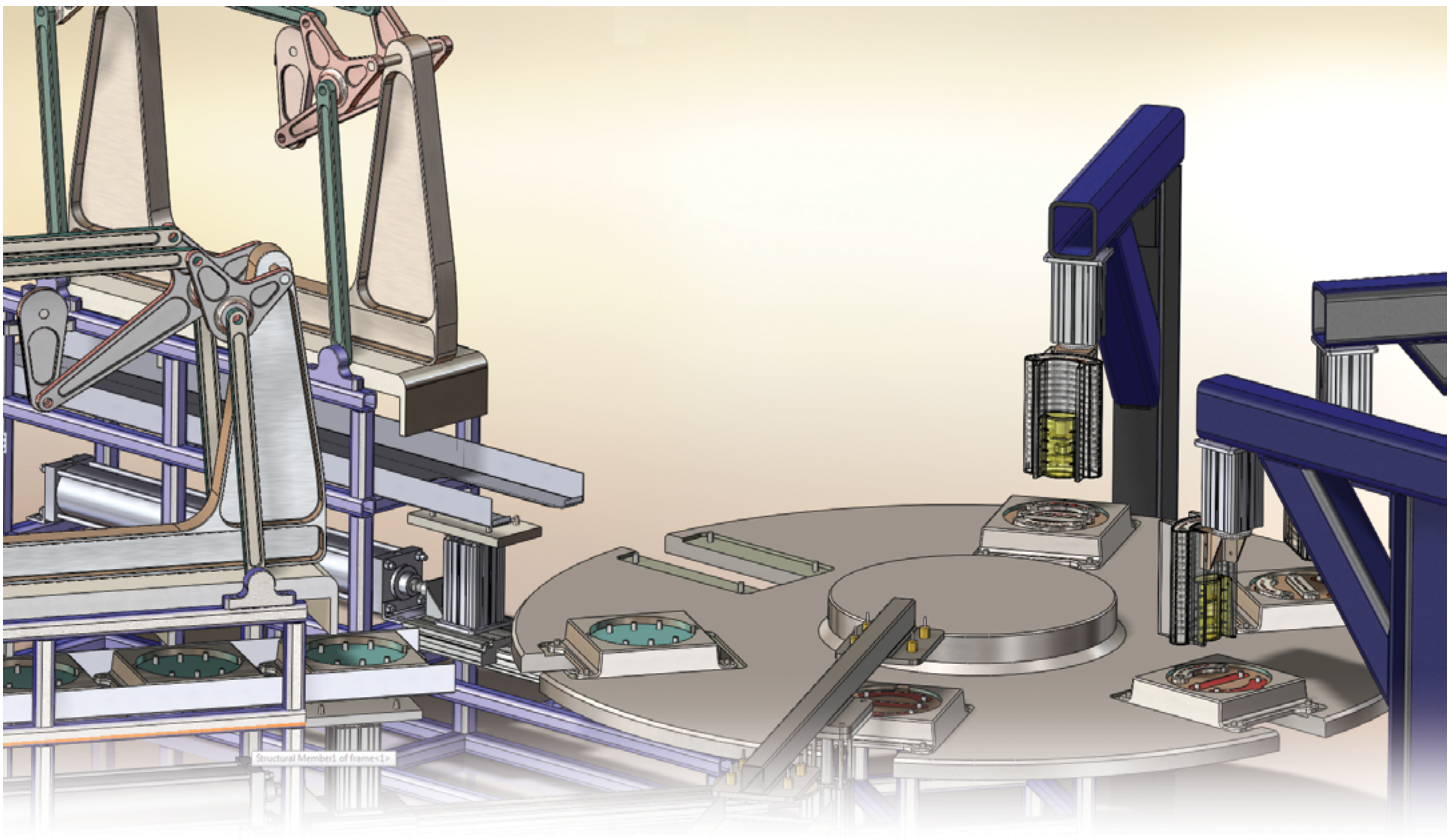
EVENT-BASED SIMULATION:
PUT YOUR DESIGNS IN MOTION

Overview

SolidWorks® software helps you move through the design cycle smarter. With flexible event-based simulation, your team will be able to integrate design and controls—and streamline your live prototype and testing process.



Introduction

Most likely, you are familiar with the fantastical machines drawn by cartoonist and engineer Rube Goldberg. These overly complicated devices accomplish simple tasks using pulleys, levers, balloons, rolling balls, and countless other mechanisms. Goldberg imagined a 16-step toothpaste dispenser and a 13-step, head-mounted, self-operating napkin. Milton Bradley's Mouse Trap® game—inspired by Goldberg's work as seen in Figure 1 below—is another favorite example in which a series of wild events catch mouse "players" in a trap.

Figure 1. A model of the classic Mouse Trap game

Manually changing timing can be tedious and difficult.

In these designs, each event is completely tied to its predecessor. If one event takes longer or shorter than anticipated, the rest of the events will adjust accordingly like a row of dominoes. This is called an event-based series.

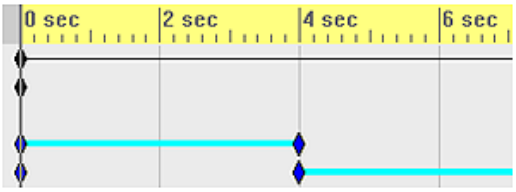


Figure 2. A mock-up of time-based video software.

There are also time-based series. Imagine you have taken a series of video clips and are trying to put them together into a movie. Your first scene is four seconds, and the next is 20 seconds, as shown below in Figure 2.

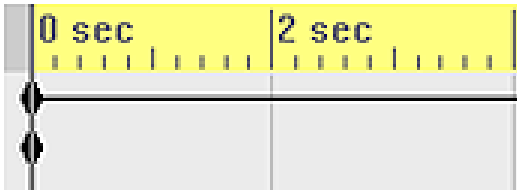


Figure 3. A mock-up of time-based video software with one second of "nothing"

After putting the scenes in a series with one another, you realize that you want to clip the first one because you only need the first three seconds. You clip it, and then…

As you can see in Figure 3, there is one second of "nothing" in between the segments. This means you may have to go back and manually adjust each of the subsequent segments to begin and end at the right time. This is tedious—in fact, if you change the timing too much, you might consider starting over.

SOLIDWORKS

Now, rather than playing Mouse Trap or creating a video, consider how these two approaches can have a fundamental impact on the simulation of a mechanical system. In this paper, we will discuss both the time-based and event-based design methods using a simple system of actuators shown in Figure 4 below.

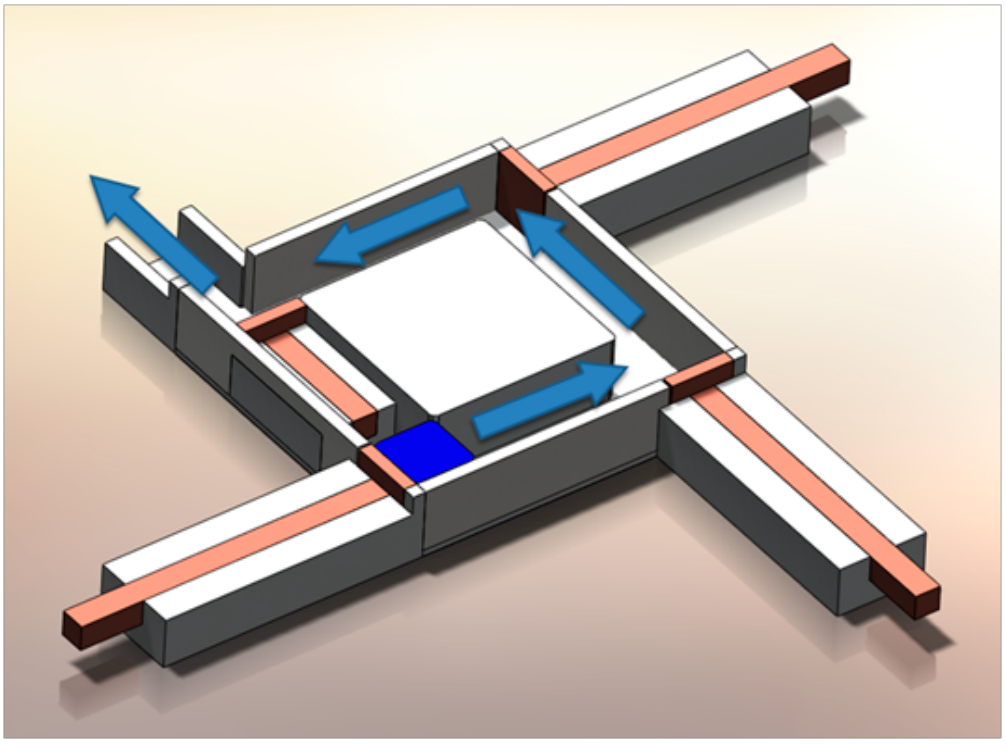


Figure 4. A system of four actuators and a blue block (left) performs a set of actions

A blue block begins in the bottom left corner and traverses the maze according to the sequence shown. Each of the four actuators pushes the block and retracts. Ultimately, the block is pushed out of the maze, and each actuator returns to its original position.

To investigate the mechanism's correct function, we can use motion simulation to virtually test the design and operation of the actuators. The motion simulation can either use a time-based or an event-based approach.

Understanding the difference between time-based and event-based simulation is important.

In order to fully understand the differences between time-based and event-based simulation (EBS), we must address their roles on two levels: the "meta" level and the "minute" level. The former considers the process holistically in the context of the machining industry's methodologies, while the latter focuses on how the process differs and how the changes may benefit each step.

The Typical Process

Machine designers typically follow the meta process shown in Figure 5 below.

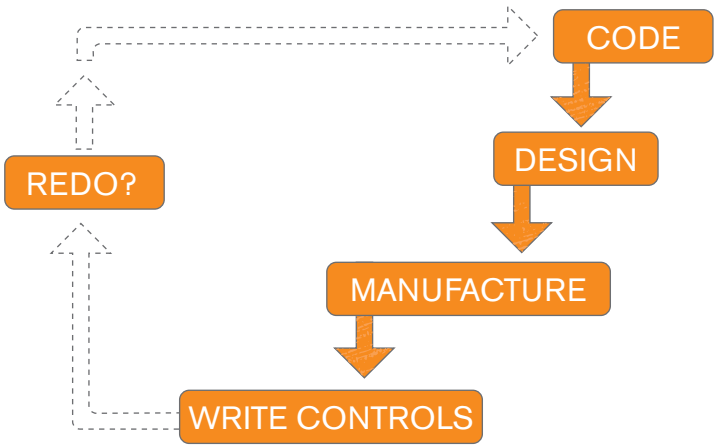


Figure 5. The standard meta design process used by machine designers

Machine designers typically follow a meta process of code, design, manufacture, write controls, with a possible redo.

Designer will often begin by writing preliminary code or layout for their machines. These codes are not to be confused with controls, the codes governing each independent mechanism's behavior; the controls link the mechanisms together. After writing these codes, the machine is mechanically designed. Once it is close to its final configuration, the design may be virtually tested, and the controls algorithms that "glue" the pieces together can be written. The controls portion of the design generally can be verified after the manufacture and assembly of the new machine.

Consider the actuators example from Figure 4. A designer's first step would be to write coding that forces each piston to move forward and then retract. This allows each individual piston's motion to be realized. The next step would be to design each of the structures in the assembly, typically using some form of CAD. The block, maze, and pistons could then be manufactured, and controls algorithms could then be written to put each piston's motion in sequence. But what if controls codes could not accommodate the timing requirements and the actuators' mechanical capabilities? For example, what if the actuators could not accelerate quickly enough? You might have to start over with either a mechanical or controls redesign.

Now we reach the "minute" level of the traditional method versus the EBS method. Generally, manufacturing processes run on absolute time, and many controls infrastructures also mandate absolute time. However, early in the design process and during simulation trials, there are significant problems with the absolute approach. You know what you want to happen, but you are unsure of exactly when each action will occur. Imagine you want to stop the first piston at "1.5 seconds" instead of the original "1 second"—an understandable tweak so early in the process, when you are still determining important mechanical details like maximum actuator acceleration. In software, the second actuator will still be set to extend at its original "1.1 seconds." This could be a problem unless you manually change it to read "1.6 seconds" to keep the original time lapse consistent. Of course, this means manually changing every step. Imagine the time required to completely alter a cascade with hundreds or thousands of steps! Worse still, if these altered controls codes become too incompatible, a complete redesign may be required, costing still more time, money, and resources.

The New Process

Clearly, if these types of alterations could be made earlier in the process in Figure 5, companies could avoid the disastrous scenario of starting from scratch. EBS addresses this need on the meta level and on the minute level. The diagram in Figure 6 below shows how the overall process changes with this feature.

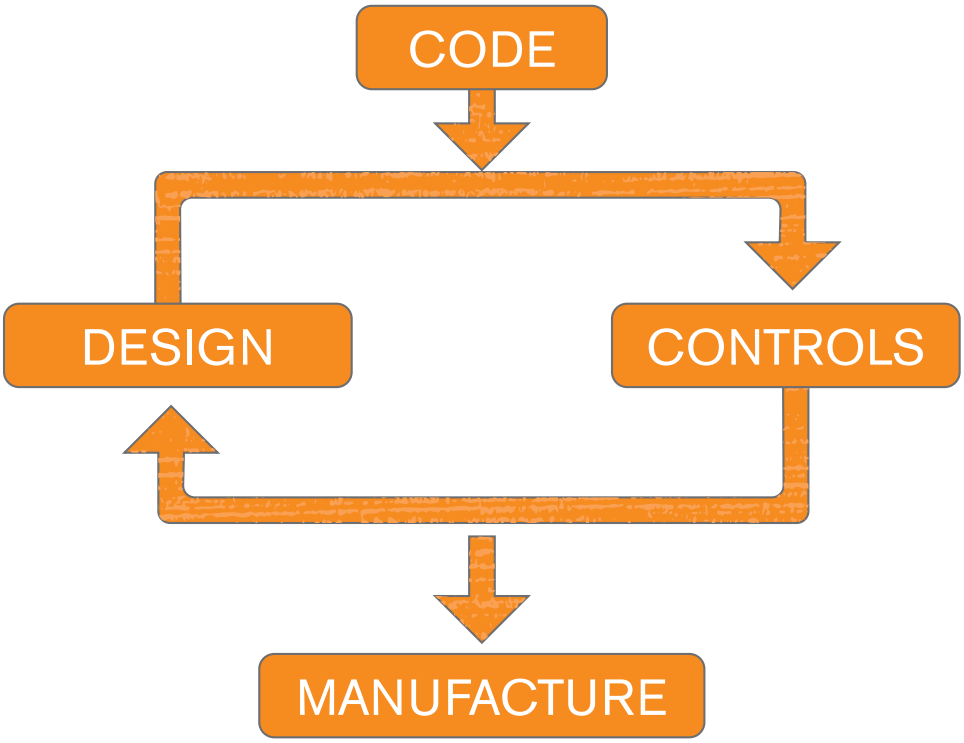


Figure 6. The new machining design process afforded by Event-Based Simulation

The initial coding procedure remains unchanged, but the rest of the process is completely reworked. You can alter the design and make a corresponding operational change, or vice versa, at the same time. This cycle means that changes and iterations can be done far more quickly than with a linear process. Finally, manufacturing becomes the last major step in the process after the design and controls procedures have been completed. Additional modifications or conversions of controls may take place afterward, but designers are far less likely to start anew than in the traditional sequential process.

With event-based simulation, machine designers have a whole new process that is easier and saves time.

On the minute level, EBS allows you to create simulations where actions are triggered by events instead of specific times. This is not meant to be exclusive, of course—the software still allows you to make simulations based on time. Figure 7 below shows a sample of the interface that governs the behavior of the four actuators from Figure 4.

	Tasks		Triggers		
	Name	Description	Trigger	Condition	Time/Delay
	Task1	Extend actuator1	Time		0s
	Task2	Extend actuator2	Task1	Task En	<None>
	Task3	Retract actuator1	Task2	Task En	<None>
	Task4	Extend actuator3	Task2	Task En	<None>
	Task5	Retract actuator2	Task4	Task En	<None>
	Task6	Extend actuator4	Proximity1	Alert On	<None>
	Task7	Retract actuator3	Task6	Task En	<None>
	Task8	Retract actuator4	Task6	Task En	<None>
	Task9		Task8	Task En	<None>
	Task10		Time		0s

Figure 7. The SolidWorks Event-Based Simulation interface to control the four actuators example.

The simulation starts with a time trigger, which can be equal to or greater than zero seconds. The next four tasks (or actions) are triggered to occur after the previous one finishes. For example, "Task 1" indicates that the second actuator extends when the first one stops extending. The subsequent actuator retractions and extensions follow a similar pattern. The final trigger to move the blue box out of the maze is the proximity sensor shown in Figure 8 below.

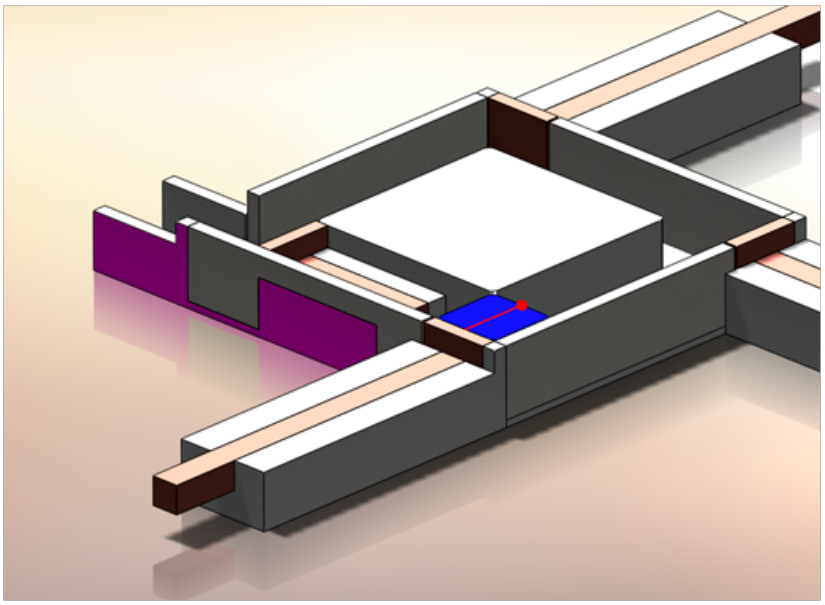


Figure 8. The proximity sensor (red) triggers the final actuator when it is within 49.00 mm of the purple wall

The proximity sensor, in red, is placed on the far side of the block from the purple wall. The sensor produces an "alert" when the is less than or equal to 49.00 mm from the purple wall. This alert, in turn, triggers the final actuator to extend outward and push the block, as shown in Task 6 in Figure 7. Note that this happens only when the block goes through the rest of the maze, at which point it becomes adjacent to the purple surface.

The SolidWorks EBS interface also includes a Gantt chart, as shown in Figure 9, to help you visualize the timing and duration of each event.

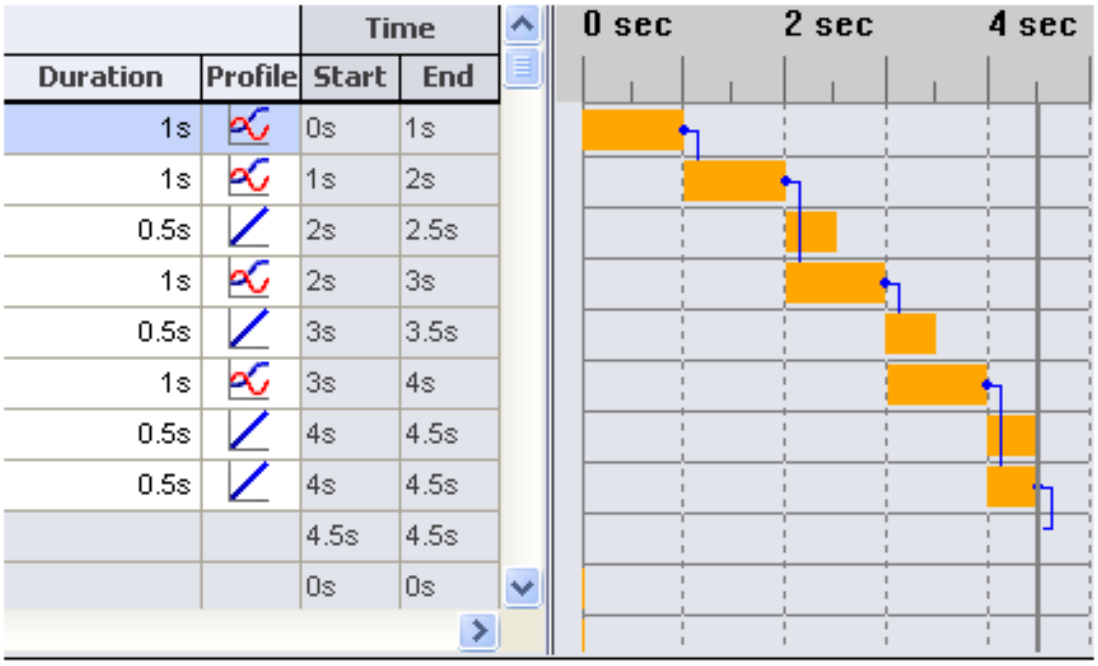


Figure 9. The Gantt chart inside the SolidWorks EBS tool.

The orange regions above correspond to event duration: the longer the orange bar is, the longer it lasts in the simulation. The thin blue lines indicate that an event is directly linked to the completion of another event (i.e., the two are event-based). The direct link between the numerical area on the left and the Gantt chart on the right allows for an instant visual presentation of the system's behavior.

Benefits of Event-based Simulation

Above all, EBS clarifies a product's design intent by significantly improving communication between designers and engineers, and controls experts. Without EBS, the two groups may speak entirely different languages. Engineers and designers try to describe their goals, but they have no sense of what controls codes require or entail. They may say, "I want this arm to turn in five seconds," but they do not know how that can be done. Controls engineers may not be clear on a proposed series of events unless it is in a coherent, code-specific form. They might reply to those engineers, "Where do you want it to turn? How does its movement relate to the rest of the system?" Using EBS tools allows designers, who are generally unfamiliar with controls design, to clearly describe their machine design goals, while at the same time controls engineers can easily incorporate these visual notes into their specific code.

On the meta level, EBS helps companies save significant amounts of time and money. The integrated nature of EBS allows for instantaneous feedback on the mechanical design as well as the proposed functionality. Companies no longer have to manufacture the product before debugging the preliminary controls algorithms. The prospect of a complete redesign is reduced as they can virtually test the machines in CAD beforehand. Furthermore, the straightforward EBS interface means that more people within the company can take a significant role in their products' designs. Engineers do not need a controls background to help define the outline of the controls requirements.

Event-based simulation may be applied to a wide variety of scenarios, including machining and more.

There are other noteworthy benefits on a more minute level. First, the event-based approach is more realistic and applicable to real life. Sensors do exist—they are vital components in many machine designs. The combination of event-based and time-based simulation that SolidWorks EBS provides is also far closer to real-world applications than an entirely time-based approach. Consider a mechanical car wash, for example. After the car enters the building, the next step in the process isn't based on time: the driver moves forward, and a sensor indicates the car is in position to be washed. It may then be washed for a set amount of time. That's an event-based controls mechanism followed by a time-based one. Countless other processes rely on a mixture of the two types as well.

Again, on a minute level, EBS makes changes much easier than the traditional method. Increasing the time on Step 1 does not mandate a cascade of manual changes through Step 95: the events and the links between them stay intact. This is important, considering the phase of the design process in which users conduct simulation: edits in controls and timing will be just as frequent as edits in CAD mechanical design. Moreover, designers and engineers are often well aware of the limits of individual parts. For example, imagine that actuator type A can extend one meter in 0.5 seconds. Replacing it with actuator type B means the extension will take 1.5 seconds. In this stage of development, laborious code changes are uncalled—EBS provides a solution.

Finally, EBS may be applied to a wide variety of scenarios. Machining operations are obvious, since the controls considerations that designers address through EBS are the same ones that coders must consider. But the simulation opportunities provided by EBS have much greater reach. Consider the field of water slide design. These designers know that the frictional forces exerted on their riders have to change over the course of the slide. Low friction is preferable at the top of the slide since it allows riders to slide very quickly. But designers must also make sure that riders slow down to avoid a painful exit, so they ramp up surface friction toward the end. Forces modeled in EBS could tell them whether their timing is right—all without major coding! Clearly, EBS is not exclusive to machine design.

Conclusion

EBS revolutionizes the way controls and CAD modeling work together. This approach allows you to directly integrate your models with the behaviors you desire, bringing mechanical and electrical design to the forefront while maintaining machine controls flexibility. EBS accomplishes this in a novel way: instead of defining specific times for events, you can start and stop each event in relation to one another. This translates into major savings. Because companies can make and modify products in simulation rather than after manufacturing, they can avoid a costly rework of the design process.

The event-based approach also saves simulation time, as it prevents a cascade of discrete time changes from bogging you down. A change in Step 1 will cleanly transition all the subsequent events. Most of all, EBS lets you communicate effectively with controls experts in many different situations. The language barrier is gone, so everyone can work together more efficiently to design better products.

You will find more ideas and help on the SolidWorks website at **www.solidworks.com.**

Dassault Systèmes
SolidWorks Corp.
300 Baker Avenue
Concord, MA 01742 USA
Phone: 1 800 693 9000
Outside the US: +1 978 371 5011
Email: info@solidworks.com
www.solidworks.com