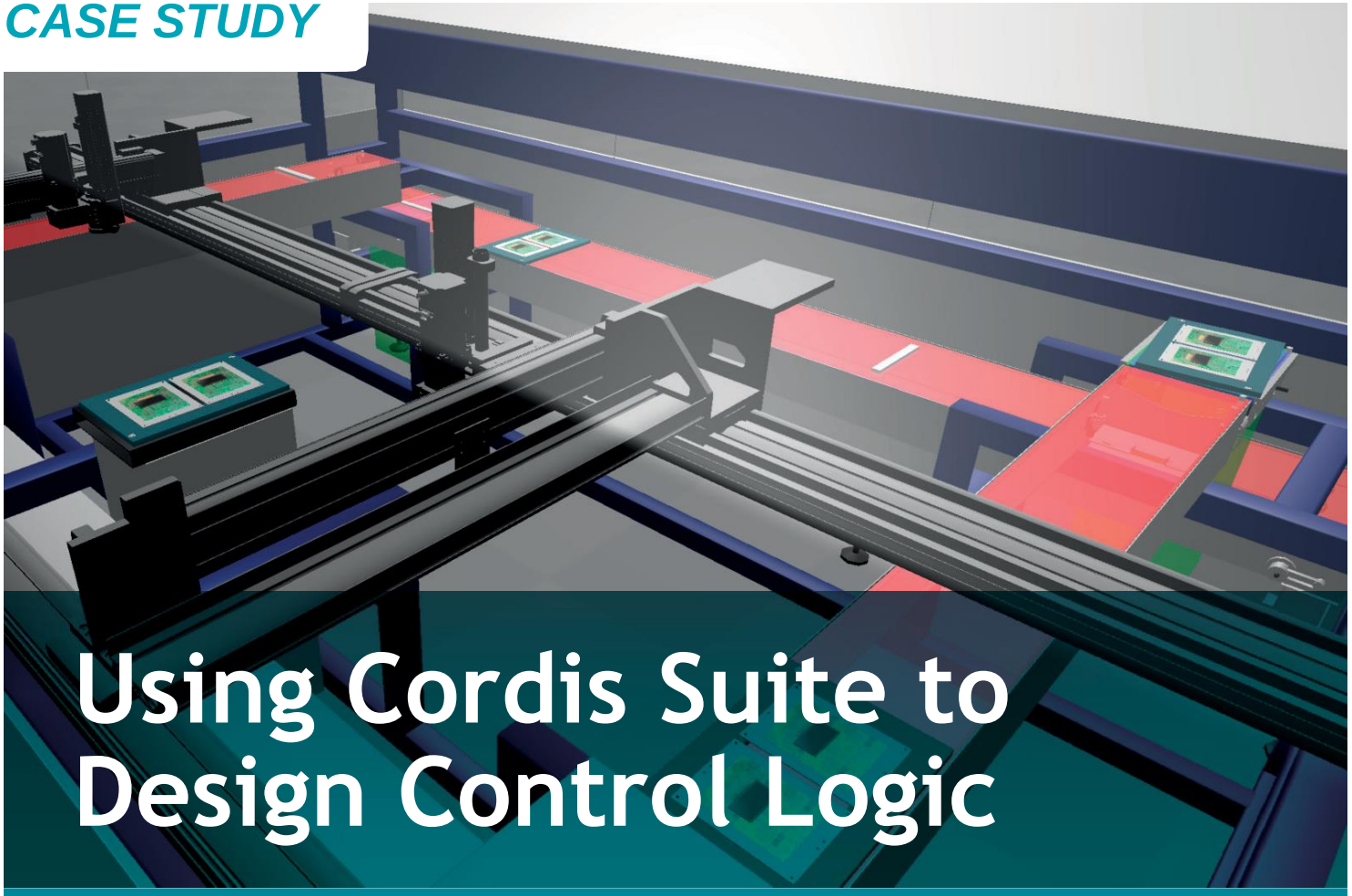
CASE STUDY



Using Cordis Suite to Design Control Logic

Cordis Suite is a practical toolset that enables you to design software according to the principles of Model-Driven Engineering. Cordis Suite is suited mainly for control logic, i.e., software used to drive moving ('mechatronic') systems. This includes software for high-tech machines such as medical equipment or 3D printers, along with infrastructural systems such as bridges and tunnels, or climate control systems inside buildings ('smart buildings').

ICT Group uses Cordis Suite and other applications when designing software based on the Model-Driven Engineering method. We take advantage of today's new technologies to train tomorrow's software engineers.

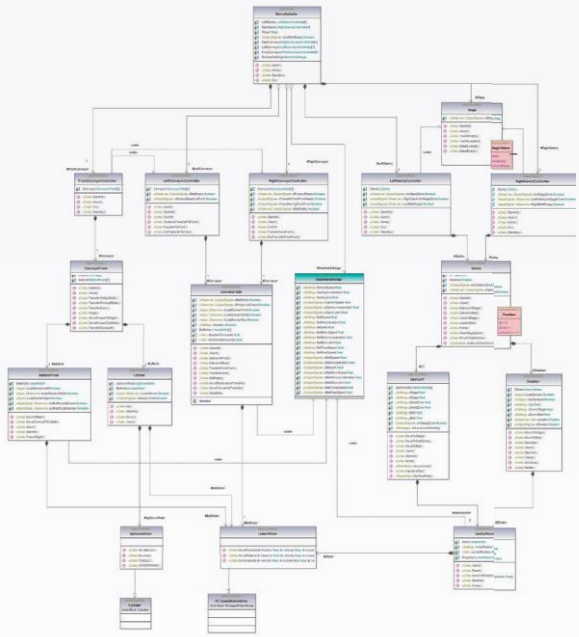
Below we describe how the Cordis Suite works in practice based on a hypothetical example: the design of software for a machine that places items on a conveyor belt using robot-operated gripper arms.

Graphic Model as a Basis for the Computer Code

The first step of the Cordis Suite is to create a functional model that describes the operation of the machine in an abstract and graphic way – this model serves as the basis for the eventual computer code. The first step is to build the static part of the model, which represents the structure of the machine, similar to a class diagram. The model establishes the relationships between the components and sub-components of the machines in a hierarchical structure.

ICT smarter solutions

The bottom of the hierarchy contains a description of the components which are similar to specific hardware components (in our example, this might be an engine or a hydraulic cylinder). The top of the model contains structures which describe more complex machine components, such as a robotic arm or conveyor belt. The static model also displays what are known as 'guards' – safety terms which restrict the functionality of the machine so as to prevent any items from breaking during testing.



Graphic representation in Cordis Suite of the static model of the conveyor belt machine.

All stakeholders are involved in the design of this model. One advantage of this model is that the software developer doesn't need to wrack their brain about the structure of the machine, but can leave this to a domain expert (e.g., an equipment manufacturer) instead. Since the model shows everything in a comprehensible, graphic design language, this domain expert can easily use it to communicate his functional requirements.

"Systems or machines are generally conceived by a mechanical engineer or systems engineer, and they then need to hand over the system to someone who will design the software," says Cordis Suite owner Benno Beuting. By designing the system in graphical terms, the engineer can communicate far more effectively with the software engineer, which reduces the number of errors."

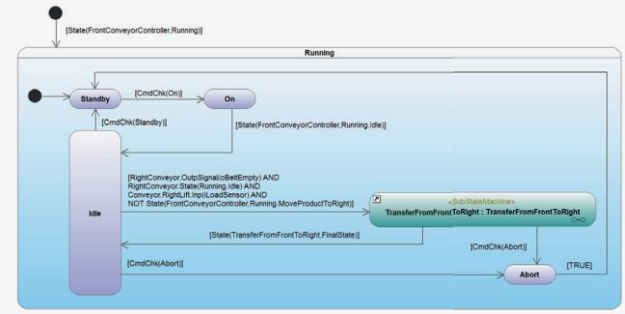
Detail of the static model that describes the relationships between the controllers of two linear robots ('gantry'.)

State Machines Reflect the Machine's Behavior

Once the static part is completed, the dynamic part of the model is designed, which describes the machine's behavior. In Cordis Suite, this is managed by state machines, which display the behavior of the various machine components. This behavior is related directly to the functional requirements for the domain. In our example, the state machines describe, among other things, which robots are active and when the conveyor belt is in motion or idle.

The dynamic model consists of different state machines which work together within the machine's structure. There are state machines for both detailed and abstract behavior. In our example, some state machines describe the motion of a single motor (in detail), while other state machines describe the operation of a whole robotic arm, in which several motors and gripping parts are integrated (abstract).

The more successful the choice of the level of detailing or abstraction per state machine, the clearer the ultimate design.



Graphic representation of a state machine in Cordis Suite. This state machine describes the activity of one of the machine's running belts. The green bar on the right is a nested sub-state in which the transfer of a product between two running belts is described.

From Model to Code with a Single Push of a Button

As soon as both models, static and dynamic, are designed to the satisfaction of all stakeholders, the computer code can be generated. What makes the Cordis Suite unique is that this step is fully automated. With a single push of a button, all functionalities described in the shared model are converted to code, and there is hardly any need for manual programming. This saves time and reduces the number of bugs in the software to an absolute minimum.

Cordis Suite owner Benno Beuting: "There is no programmer in the world who can produce completely error-free code; you generally need to look for and solve programming errors after the fact. The great advantage of Cordis Suite is that it automatically converts the design into error-free code."

In converting the model to code, there are several different programming languages to choose from. Cordis Suite supports several commonly used PLC languages, along with programming languages such as C#.

Only data processing or the creation of 'glue code' (e.g., in order to embed MDE-designed software in a larger software design) needs to be programmed manually.

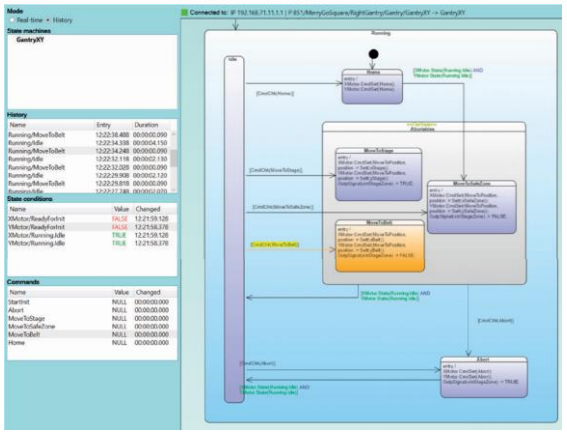
There is also a need for configuration, which involves linking the artifacts from the generated code to the physical machine components. In other words, the software IO must be manually linked to the hardware IO, an operation that generally only needs to be performed once.

Modifications at Any Time

This automated code generation facilitates a significantly shorter iteration cycle. The software designer and the stakeholder(s) receive feedback significantly faster and can immediately see the impact of their changes. At the same time, it is also possible to implement changes in the software design at a very late stage.

If our hypothetical machine were to demonstrate, for example, that an additional sensor is required in order to streamline the process, this can easily be added to the static model. This is then flawlessly converted into new code with a single push of a button, thereby eliminating the need to go through the entire code and keep making manual corrections along the way.

The machine can be operated using the Cordis Suite dashboard; this applies to both the machine as a whole and the separate components. The dashboard shows the machine's behavior live through the state machines, along with the 'state history': a list that shows the user at a glance what tasks the machine has completed.

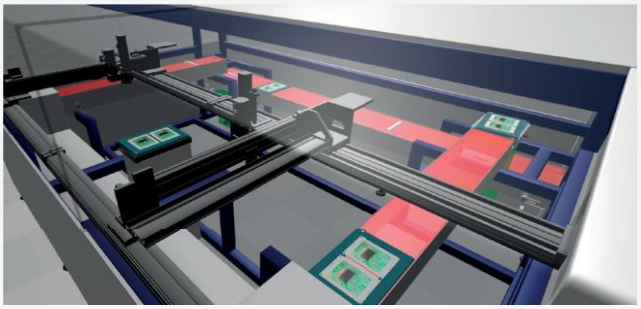


In Cordis Suite, the dashboard (pictured on the left) serves to operate the machine. The right-hand side shows the dynamic model, whereby the current activity of the state machine is highlighted in yellow.

Fast and Easy Testing Using a 'Digital Twin'

A unique feature of Cordis Suite is the option to test the software on a virtual machine, known as a 'digital twin.' This virtual copy of the physical machine makes it possible to test and adapt the software at an early stage: this is a marked difference with the traditional method of software design, where tests are performed only toward the end of the development process. Early testing helps to identify errors faster and at an earlier stage, making it possible to correct them, instead of cutting it close to the deadline, as is customary in traditional software design.

An added advantage is that the tests are not necessarily conducted on-site, since a virtualization feature is available. This allows designers to operate independently of the availability of the eventual machine, which is often still under development. The software is used to run the tests, with all stakeholders being given the opportunity to follow the process remotely online and provide feedback. Even if there are remaining issues following delivery, this can be communicated and resolved using the virtual machine, which is essentially a digital mock-up of the eventual physical machine. This mock-up can immediately be communicated to the client, so that the latter can get an idea of the end result at an early stage.



The 'digital twin' of our (non-existent) conveyor-belt machine. This virtual simulator reacts in the same way as the ultimate physical machine would.

The Main Differences with Traditional Software Design

The Cordis Suite method varies from the traditional software design method in two key areas. The main difference is that the computer code is automatically generated by the functional model; what this means in practice is that the process of designing the model takes up the largest amount of time and attention. The design stage therefore takes longer than with the traditional method, but this is more than offset by the automatically generated code, as the programming is instantaneous. This new design method also requires a new mindset, but it ultimately leads to greater savings in terms of time, money, and efficiency.

The second improvement is the use of simulated hardware – also known as the 'digital twin' – which makes it possible to conduct tests and receive feedback at an early stage. Stakeholders can view the impact of changes in the model directly in real-time. Moving the test stage forward also reduces deadline pressure.

If you would like to find out more about the options provided by Model-Driven Engineering and Cordis Suite, please contact our software designer, Olaf Donk, to discuss the options available for your organization on a no-obligation basis.

ICT smarter solutions

ICT Group
Kopenhagen 9
2993 LL Barendrecht,
The Netherlands

T +31 (0)88 908 2000
F +31 (0)88 908 2500

info@ict.nl
www.ict.eu