A case study of multi-disciplinary modeling using MATLAB / Simulink and TrueTime[1]

P.F.A. van den Bosch
Océ Technologies BV
PO Box 101
NL-5900 MA Venlo, The Netherlands
fvdb@oce.nl

E.H. van de Waal
Imtech ICT Technical Systems
P.O. Box 1608
3800 BP Amersfoort, The Netherlands
evert.vandewaal@imtech.nl

Abstract. When developing a complex machine, designers have to make design choices with far reaching consequences for the performance of the system. Often those choices are made without being able to properly foresee these consequences, because of lacking knowledge.

In this paper, it is shown via a case study that by using multi-disciplinary models, designers can obtain approximate results that are very helpful in evaluating a design choice.

A model of the paper flow in a high volume printer / copier is made. This model focuses on timing and electrical power consumption. The model is validated using measurements on a real printer. It is shown that the model can be used to evaluate the impact of design decisions on the system.

I. INTRODUCTION

Designing a complex system

The complexity of products being designed by industry today is increasing at an astonishing rate. The search is for a product that will satisfy the requirements within certain margins, e.g. a certain development cost, production costs, speed of operation, time to market, functionality, physical dimensions, power consumption, noise production and so on. Often, these requirements are conflicting so that a right balance must be found.

For a system, multiple disciplines need to make designs, e.g. the electronic design, mechanical design and software design. Designs are often made in parallel by multiple groups of people. In this process, often choices are made which may have benefits in one discipline but disadvantages in other disciplines. When is such a choice a good choice and when not? To answer that question, the overall effect of such a choice needs to be evaluated. The earlier a problem is found, the easier it is to repair. Thus, we need a method to evaluate a design choice as early as possible over multiple domains.

Often, prototypes and modeling are used for this purpose. Physical prototypes are essential because of the confrontation with physical reality, where overlooked areas are made clear. However, it is difficult to quickly evaluate different designs through physical prototypes because a new prototype is needed for each design. Only through evaluation of models can different designs be compared quickly, because of the ease with which a model can be modified and re-evaluated.

[1] This work has been carried out as part of the Boderc project under the responsibility of the Embedded Systems Institute. This project is partially supported by the Netherlands Ministry of Economic Affairs under the Senter TS program.

Model evaluation can take many forms: from back-of-the-envelope calculations, to simulation of elaborate models incorporating many of the details of the design. This paper will show our approach to modeling a complete system, where high-level control drives a simple model of low-level control and mechanical behavior.

Importance of system timing

In complex products, the design will generally include mechanical, electrical, control theory and software elements. When building such models, a binding factor needs to be found. For physical elements, time and energy are common denominators. This is recognized by some simulation environments, e.g. Simulink, 20-Sim, Modelica and Ptolemy, which have a modeling paradigm based on time and power flow.

In a system, timing is usually of critical importance. System timing has a large impact on e.g.

- Throughput.
- Power consumption.
- Noise production.
- Bill of materials.
- Product value.

Thus, in order to evaluate a design against the initial requirements, it is essential to model the timing of the whole system.

It is stated in (Bode-RC, 2002) that a good way to weigh design choices over different disciplines is to build a common model. The problem is how one can model both the physical world of electronics and mechanics as well as the software world.

The system under study

The printer under consideration is the Océ VarioPrint 2090. A model has been made of the paper path of this printer, predicting the timing of paper movements and the required electrical power. The model includes motors, pinches, and sensors, all with their (mechanical) positions on the paper path. Also, the control software has been modeled.

Figure 1 shows the printer and its components. This article focuses on the paper path, including the Paper Input Module (PIM), the Registration Module (RegMod) and the Finisher. The RegMod is responsible for delivering the sheets in time at the fuse-point, where the image is printed onto the paper. The image is first developed on the OPC (Optical Photo Conductor), then transported via the TTF (Transfer Transfuse), and finally fused on the paper.

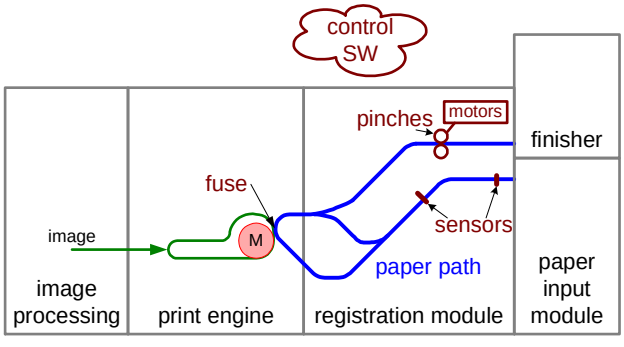

Figure 1: Picture of the Océ VarioPrint 2090, with a schematic representation of the different components.

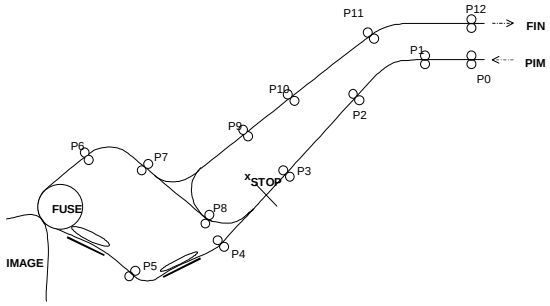

Figure 2: The paper path, showing the positions of the pinches and the bypass and duplex loop.

There are four embedded controller boards in the printer/copier: one for the ADF/scanner, one for the RegMod, one for the finisher and one for the print engine, which also controls the PIM. The controller boards communicate through a CAN network. The print job controller controls the overall operation.

The main part of the paper path is the RegMod, therefore the RegMod is modeled in most detail. The mechanical layout of the RegMod is shown in figure 2. Sheets enter the RegMod from the PIM. Then they are stopped at the stop position in order to be aligned. After being aligned, they proceed to the fuse and then either to the finisher (simplex) or back to the stop-position to be printed on the other side (duplex). Some sheets are not printed at all: they move directly from the stop position to the finisher (bypass).

Modeling approach

The goal of the model is to capture the timing of paper flow through the paper path of a printer. A second goal is to model the power usage of paper transport, so that it can be minimized. The model includes the mechanical layout, e.g. the distance between pinches and the pinch diameters. Furthermore, the motors driving the pinches are modeled in both the electrical and mechanical domain. Sheets have a certain length, and cover optical paper sensors depending on their position. The software that controls the paper flow is also modeled: it reads the paper sensors and then decides on the set points of the motor controllers.

The simulation package MATLAB / Simulink of (The MathWorks 2004) is used to model the mechanical and electrical effects (sheet, pinches and motors). Simulink is extended with the library TrueTime to model software behavior. (Henriksson, 2004) proposes an extension to Simulink that enables the simulator to model the behavior of processor blocks running an RTOS (Real-Time Operating System) and communication channels between those processors. This toolbox was developed to study the effects of distributing a control system over several processor nodes. However, in this article we show how it can be used to model the higher-level control software that controls an entire system.

TrueTime is implemented as a Simulink block, which can easily be combined with other Simulink blocks (Henriksson, 2002). The execution of each task is divided into several segments which all have a specific execution time. The modeler can specify this duration in the code. Each segment is executed atomically, after which the TrueTime kernel checks whether the task has been pre-empted by another task or can continue running. The simulated time at which a segment executes is determined from the duration of the preceding segments.

Furthermore, TrueTime simulates a network connection through which multiple processor blocks can be interconnected. The communication delay of the communication link is also simulated.

Using TrueTime, not only can functionality of software be modeled accurately, but also the timing behavior. This is not possible with e.g. StateFlow (MathWorks, 2004a).

II. MODEL

The model consists of a physical part (motors, pinches, sensors etc.) and an embedded control part (processors and communication buses). In figure 3 the top view of this model is shown, the output of the embedded control is input for the physical model and vice versa.

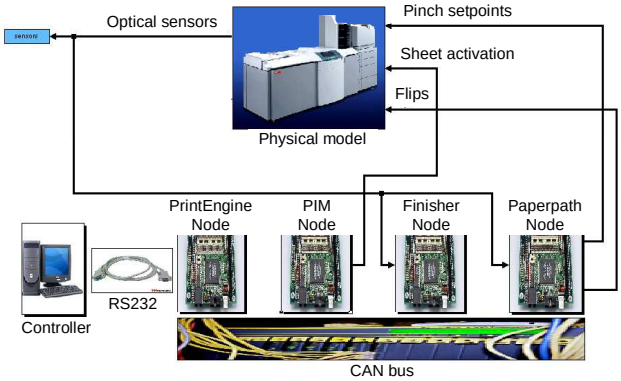

Figure 3: Highest level view of the printer model: an electrical/mechanical system is controlled by four embedded controllers and one job controller.

Modeling sheet movement

Sheets are modeled as integrators, and because they are assumed to have zero mass, the integration of the speed of the driving pinch gives the sheet position. In order to know the speed of the sheet, it must be known which pinch is currently controlling the sheet. For this, the sheet position is compared to the position of all pinches and sensors. When a sheet enters a pinch, that pinch takes control over the sheet. If a sheet triggers an optical paper sheet sensor, a signal is generated that is used by the software model.

For the simplex case, the paper path can be seen as a simple straight line. However, in the duplex case, this is no longer possible. To deal with this problem, the paper path is divided into segments, as shown in figure 4. In the duplex case a sheet will move from segment 2 to segment 5 and then segment 9, where segment 9 is actually the same as segment 1 except that the orientation of the sheet is reversed. The same duplication of segments is done for segments 0, 2 through 7, but that is not shown in the figure for reasons of clarity. Whether or not the duplex path is taken, depends on the state of the switches (flips) at the crossings of segments, indicated by the circles in figure 4.

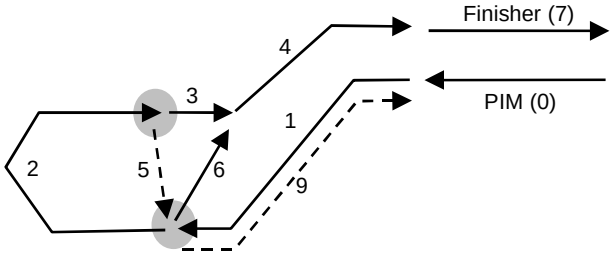

Figure 4: The division of the paper path in several segments: The arrows indicate the positive direction; dashed arrows are for duplex. The circles represent the flips.

The Paper Input Module (PIM) and finisher are considered as two separate segments, which are not modeled in detail but are necessary to start and finish sheet movement.

Using this scheme, the sheet position consists of two coordinates: the position within a segment, modeled by the value of the integrator, and the number of the segment it is in. So, the position can be identified by (p,s), with $0 \leq p \leq \text{length}_{\text{seg(s)}}$ and s one of the segments (0..15). When a sheet enters a segment, the integrator is set to zero.

Mechanical Model

Several simplifications have been made in the mechanical model. The motor model does include Coulomb and viscous friction, but losses and friction between pinches and sheets are neglected. If a sheet is in several pinches at the same time, it gets the speed of the pinch closest to the leading edge. This simplification is in most cases justified because of the usage of uni-directional bearings. In the cases that it is not a valid assumption, the error is not very large, but to get more accurate results, the model has to be changed to incorporate the effect. Interaction through sheets between several pinches is neglected.

The model includes several motors, each having different parameters. The parameters have been determined experimentally in earlier work (unpublished), and apply to a warm printer.

Electrical model

The electrical model of the motors contains resistance and inductance. Each motor has its own amplifier that share a common voltage source. These amplifiers, H-bridges, are assumed to be perfect, delivering a controllable percentage of the supply voltage to the motor. The voltage source is modeled as a constant voltage with a linear resistor simulating the voltage drop due to the total current drawn by the motors. Therefore, there is electrical interaction between the motors.

Software model

In our model (figure 3), the software is distributed over four processors that communicate through CAN. The PrintEngine node is the main node; it includes the management software that tells other nodes what to do. The PrintEngine node is the one that plans how and when the sheets go through the paper path and when the images are generated. The other nodes make sure that this planning is carried out. This planning and also notifications from PIM to RegMod and to the finisher are communicated using the CAN bus.

In the real printer, almost all nodes run motor controller loops, with a frequency of 500 Hz or 1 kHz. Modeling this in TrueTime would require the model to be evaluated 1000 times per second, which causes very slow simulations. By simulating the motor controllers in continuous time, the number of model evaluations is reduced.

Simulation experiment

The model was evaluated for simplex A4 sheets being printed at maximum speed (85 PPM). The resulting position / time diagram for the simulated sheets is shown in figure 6. This experiment took approximately 30 minutes on a 1.6 GHz Pentium-4 processor, to simulate 10 seconds of system operation, including the simulation of motor and controller behavior. The simulation results, including power usage, will be discussed further in the following sections. Note that the dynamic behavior of motors and controllers has been taken into account, as can be seen from the resulting overshoot of one of the motors and therefore the sheet position.

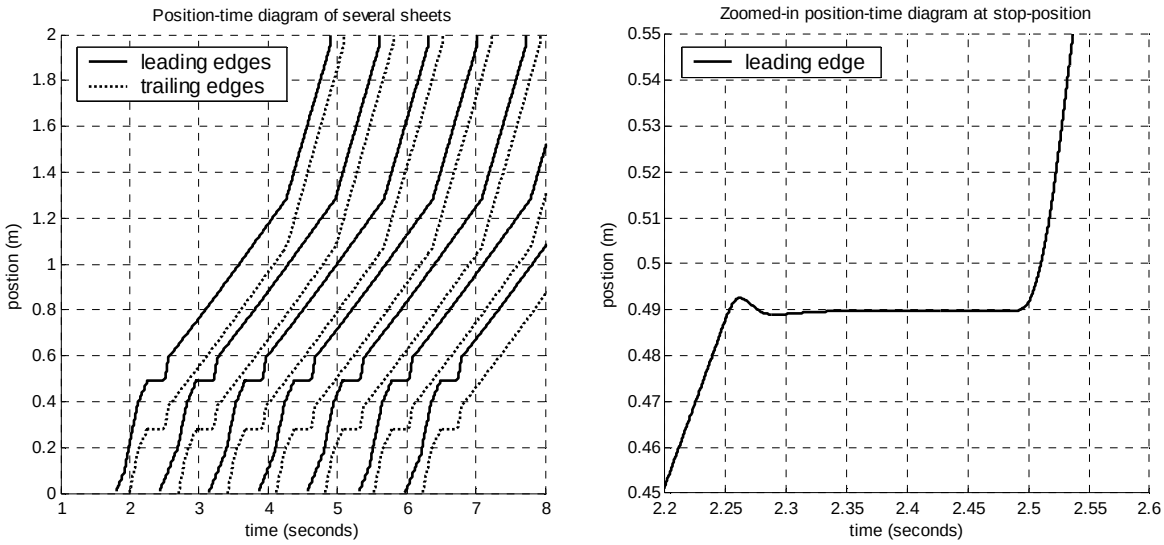

Figure 6: Simulation result showing position of paper edges over time, for simplex A4 (left) and zoomed in at stop position (right).

Remark: During the creation and testing phases of the model, the slowness of execution can become a bottleneck. Therefore it is possible to bypass the motor and controller submodels and set the desired set-points directly to the pinches. This way the model developer can tune and debug the model on aspects that are not influenced by the *exact* motor behavior. Doing the simulations without motor and controller models results in 14 seconds to simulate 10 seconds, which is much faster.

III. VALIDATION

In order to validate the power behavior of the model, the current through and voltage over one of the motors is measured, while printing pages of A4. The measured behavior of the motor was compared with the simulated behavior, as shown in figure 7. It can be seen that the current and voltage over motor 3 were simulated quite accurately. However, the simulation of power is sometimes not very accurate. This is due to the multiplication of errors, especially for high voltages.

Some differences in the timing behavior can also be observed. For example, in the simulation, the motor starts running immediately, in reality it is started just before paper arrives. This is why there is a peak in power at t=0.57 s which is not in the simulation. Another timing difference can be observed at t=1.3 s, 2.0 s, 2.7 s and 3.4 s. This is due to a sheet correction that turned out to be implemented at a different time than documented in the design specification. Other timing differences result mainly from disturbances and parameter variations.

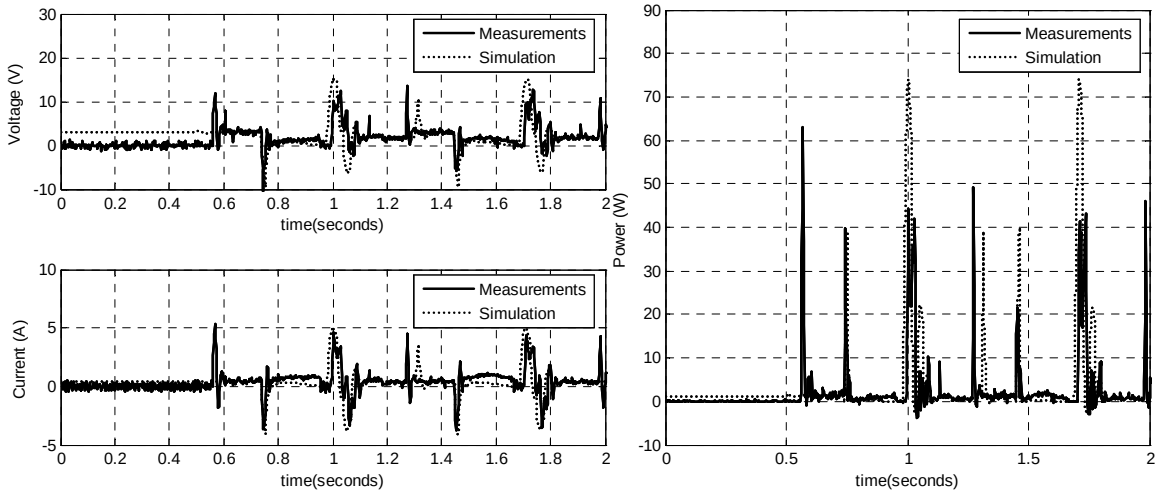

Figure 7: Comparison of simulated and measured voltage, current and power.

IV. TRADE-OFF ANALYSIS

One important feature of the system is the peak power usage. For example, by reducing the power consumption, smaller and therefore cheaper power supplies can be used. In the printer, limiting the amount of acceleration given to sheets can effectively reduce the amount of peak power used. However, this will also reduce the amount of timing deviation that can be corrected by the printer.

In a simulation experiment, the maximum acceleration has been almost halved. For both the originally allowed accelerations of 50 and 35 m/s^2 and for the lower values of 35 and 15 m/s^2, the power consumption of all paper path motors has been calculated. Leading to an original value of about 120 W, and 70 W for the lower acceleration, see figure 8.

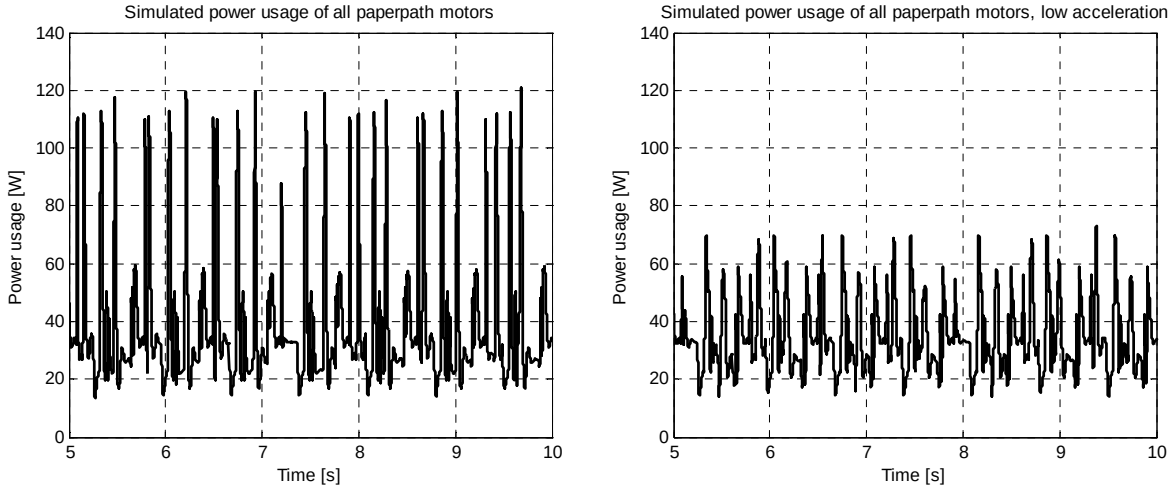

Figure 8: Simulation of power use and timing deviation correction for high acceleration (left) and low acceleration (right).

However, the ability to correct the deviations with which sheets arrive from the PIM (a variation of 100 ms) is decreased. Normally, the first motor uses a correction profile to undo

these variations. From the simulations it follows that if the maximum acceleration of this profile is reduced, the motor does not have enough time to correct the sheets when they arrive 100 ms too late. The result is that the next motor gets those sheets too late as well, which would be undesirable because that would lead to miss-alignments between sheets and images.

This result can be used to negotiate design modifications in a design team. It is shown that it is acceptable to use lower acceleration, but only if the PIM delivers sheets with an accuracy better than 80 ms. A trade-off can now be made which would be beneficial.

V. CONCLUSIONS

In this case study we have shown that it is possible to build a multi-disciplinary system level model of a printer using Simulink and TrueTime. This model has been partly validated for the power usage of a motor. It has also been shown that the model can be used to analyze trade-offs between robustness for paper timing deviations and power consumption.

There are many more parameters in the model whose effect on, for example, robustness and power consumption can be evaluated. Examples are the positions of sensors and pinches, the motor parameters like its mass, motor constant, friction etc. But also the speed of the CAN bus might become a bottleneck if it is too low. In this case study all those other parameters where not touched, but these can be modified just as easy as the maximum acceleration rate. Therefore we make the intuitive argument that using this system model is faster than building prototypes to evaluate design choices.

VI. REFERENCES

Bode-RC, *Summary of the Boderc Project Plan*, Embedded Systems Institute, Eindhoven, `http://www.embeddedsystems.nl`

Henriksson, D. et al., "TrueTime: Simulation of Control Loops Under Shared Computer Resources", *Proceedings of the 15th IFAC World Congress on Automatic Control*, (Barcelona, Spain, July 2002).

Henriksson, D., “TrueTime: Simulation of Networked and Embedded Control Systems”, Lund University, Sweden, 2004, `http://www.control.lth.se/~dan/truetime/`

The MathWorks: The MATLAB / Simulink Homepage. `http://www.mathworks.com` (2004)

The MathWorks: Stateflow. `http://www.mathworks.com/products/stateflow/` (2004a)

VII. BIOGRAPHY

Peter van den Bosch received his M.Sc. degree in Electrical Engineering from the TU/e (Technische Universiteit Eindhoven, The Netherlands), in 2001. Since 2002, he is a researcher at the research department of Océ Technologies BV. Since 2003, he is working on the Boderc project at the Embedded Systems Institute in Eindhoven.

Evert van de Waal received his M.Sc. degree in Electrical Engineering from the University of Twente, The Netherlands in 1993. After doing research on optimal adaptive control at the Industrial Control Center (University of Strathclyde, Glasgow, Scotland), he has been involved in the development of embedded software, usually machine control software. Since 2001 he has

been a consultant with Imtech ICT. Since 2003, he has been working part-time on the Boderc project at the Embedded Systems Institute in Eindhoven.