The performance of product-driven manufacturing control: An emulation-based benchmarking study

Rémi Pannequin, Gérard Morel, André Thomas

The performance of product-driven manufacturing control: An emulation-based benchmarking study

Rémi Pannequin, Gérard Morel, André Thomas

Research Centre for Automatic Control of Nancy (CRAN - UMR 7039)
CNRS, Nancy Université
F-54506 Vandoeuvre les Nancy, France

Abstract

Product-driven control may enable manufacturing companies to meet business demands more quickly and effectively. But a key point in making this concept acceptable by industry is to provide benchmarking environments in order to compare and analyze their efficiency on emulated large-scale industry-led case studies with regard to current technologies and approaches. In this paper, a benchmarking protocol is defined, in order to provide R&D practitioners with benchmarking services in a product-driven implementation project. A component-based generic architecture is proposed to support this protocol, enabling to model and compare various control architectures. This benchmarking protocol is applied to an automotive-industry case study in order to evaluate the impact of making the products interact with the local decision centers. Finally the experiments show that product-driven control can perform as good as traditional centralized control, and that its robustness depends mainly of the local decision-making processes.

Key words: Intelligent Manufacturing Systems, product-driven control, benchmarking, simulation, HMS, MAS

1 Introduction

There is a large consensus in the IMS community between holonic control, production management and virtual enterprises (Babiceanu and Chen, 2006) that the combination of agent and infotronics technologies may permit meeting flexibility and adaptability issues as required by the increasing customization of goods and services. Indeed, on one hand, agent technology brings forward new fundamental insights on decentralized coordination and auto-organization, enabling new manufacturing decision-making policies and on-the-fly reconfigu-

ration capabilities ("plug-and-produce"). On the other hand, infotronic technologies address mainly the issue of synchronization between physical objects and their informational representation, which was acknowledged to be one of the major issues in production and inventory management (Plossl, 1985).

This holonic-modeling paradigm, mainly developed by the Holonic Manufacturing Systems community (Valckenaers, 2001) can be applied into product-driven control. Product-driven control is a way to exchange the hierarchical integrated vision of plant-wide control for a more interoperable/intelligent one (Morel et al., 2007), by dealing with products whose information content is permanently bound to their material content and which are able to influence decisions made about them (Mcfarlane et al., 2003). This approach is applicable at the supply chain level to improve products information management, with applications of the system in tracking and logistics control (Kärkkäinen et al., 2003).

Research interest in product-driven control is currently growing, as numerous research projects appear. For instance the European project PABADIS'PROMISE [1], involving researchers and software vendors, aims at building a product-centered manufacturing execution system, by developing a new kinds of devices capable of embedding an agent representing a product, and ERP modules and process controllers capable of interacting with these product agents.

Improvements in observation of products, thanks to identification technologies (such as RFID, bar codes, etc...) are therefore driving major changes in the way control architectures are organized. One of the key organizational issues is business to manufacturing (B2M) interaction; holonic products integrating information from both the business and manufacturing point of view are able to solve interoperability issues (Baina and Morel, 2006). They may also make possible to have seamlessly coexisting mature centralized decision systems, such as MRP2 (Vollmann et al., 1997) with newer distributed control approaches. Figure 1 summarizes this novel organization centered on active products.

Nevertheless, as noted in recent work by Marik and Lazansky (2007), there is still a long way to go to make these heterarchical architectures efficient in real industrial environments. Among many issues to be solved, embedded devices, as well as agent technologies, are not yet sufficiently reliable and powerful to handle the scalability problems so that decision-making can be fully shared.

Another issue is then to identify the correct balance between centralized and distributed control capabilities of decision-making agents, so that the process can be digitally interactive in both directions, from the operators down to

[1] Plant Automation Based on Distributed Systems Product Oriented Manufacturing Systems for Re-Configurable Enterprises (http://www.pabadis-promise.org)

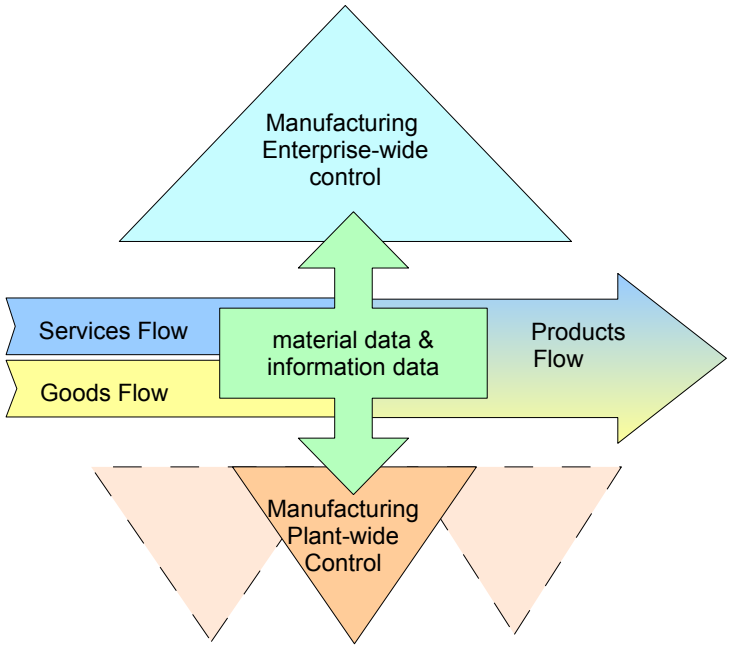

Fig. 1. B2M interaction centered on holonic products (Morel et al., 2007)

the products and back. A modeling and testing environment for analyzing and comparing alternative decision-making scenarios with traditional ones is required. The experimental feedback on design decisions provided by such a benchmarking environment would permit the firm grounding of product-driven architectures, and could discover new issues.

Therefore, this paper proposes an architecture based on generic components, which can be used to build emulation-based benchmarking models. This environment enables researchers or practitioners to easily build industry-scale executable models of factories in order to test control systems, and to model and execute various organizational schemes of the decision-making system.

After stating in Section 2 the requirements for a benchmarking environment in the context of product-driven control, we will define the proposed components-based architecture; Section 3 deals with building an emulation model of an industrial shop-floor system. This modeling and testing environment is then applied in Section 4 to an industrial case study where a manufacturing line of an automotive industry subsidiary is considered. In Section 5 we address the interpretation of the experimental results; the Conclusions section presents some perspectives into ongoing works at CRAN for industrial transfer in the area of product-driven logistics for the natural-fiber industry.

2 Problem statement

An empirical approach seems the most feasible way to evaluate new control architectures. By *empirical*, we mean that generic properties of the control architecture are logically induced from concrete applications of the control system to many particular cases. Theoretical approaches, such as evaluation of computational complexity (Monotori and Csáji, 2007) may generate useful results, but cannot cope with the complexity of distributed systems. Indeed, the behavior of distributed systems is often based on the *emergence* of global properties from local behavior and interactions. But mathematically formalizing emergence is not easy (Corning, 2002), and modeling the system as a whole may results in models more complex than the theory can actually solve. So, by using an analytical approach, researchers might have to simplify reality, or to choose only simple situations.

We distinguish three ways for conducting such concrete performance measurements: namely analytical methods such as queuing theory, Monte Carlo methods such as simulation or emulation, and physical experimentation such as lab platforms or industrial pilot implementations.

In the context of product-driven control, analytical methods are impractical because the mathematical models corresponding to a realistic case are often too complex to be solved. Physical experimentation has the disadvantage of technical- and cost-related limitations (Valckenaers et al., 1997). Simulation seems the only recourse for modeling and analyzing performances in large-scale industrial cases.

Comparison between antagonistic control modes such as market-based and hierarchical control (Cavalieri et al., 2000) or planning-based and reactive control (Brennan, 2000) have been carried out using specifically developed test beds, but more generic evaluation tools are needed, enabling us to store, share and compare test cases.

The development and definition of such generic evaluation tools has drawn a great deal of interest. An online benchmarking utility has been defined by IMS-NoE special interest group 4 (Cavalieri et al., 2003, and Valckenaers et al., 2006), and this enables collection and sharing of a wide range of industrial test cases. Until such a generic service—one is under development at KU Leuven—is available, simulation-based benchmarking of complex manufacturing systems remains the way to establish proof of the efficiency of plant-wide-control organizational strategies before their deployment for practical purposes (Monch, 2007).

In fact, there is consensus on the architecture of benchmarking environments putting the emphasis on modularity between the control system *CS* and the

shop-floor system *SF*. With these notations, an experimental run consists in verifying the assertion (Fusaoka et al., 1983) $SF \wedge CS \supset G$, where G is the required performance level. The real shop-floor system may be replaced by a model, an *emulated* shop floor. Likewise, a model of the control system can be used instead of the real one. Therefore, four experimental situations can be defined, using either models or real systems (Pfeiffer et al., 2003).

Nevertheless, modeling issues remain. For instance, it can be difficult to distinguish between the shop floor and control systems. The borderline between the two systems depends of the level of the control functions being tested. Another issue is to build the emulation model itself, as currently available simulation software does not offer emulation-capable modeling components. A third issue is to be able to develop easily used product-driven control systems, in order to compare various architectures or to validate decision-making policies.

This paper aims at solving these modeling issues by defining a modeling and testing component-based framework. Moreover, we propose two experimental steps, to enable iterative development of the product-driven execution system.

In the first step, a model of the control system is applied to the emulated shop-floor system ($SF_m \wedge CS_m \supset G$). Under the hypothesis that product-embedded data are available to decision centers, this approach can be used to define what kind of data should be embedded into products and how decision algorithms should use them. The second step uses the real distributed control system (e.g. a multi-agent system) that controls the emulated process ($SF_m \wedge CS \supset G$). This testing approach enables us to consider software-related issues such as how products and decision-making centers should interact to exchange data.

This paper focuses on the first experimental step. An industrial case study dealing with a product-driven decision scheme is presented. Previous works on this case have discussed the respective performances of centralized and distributed control (Pannequin and Thomas, 2004). Both decision modes are combined using products: the centrally made predictive schedule is used to initialize product data, whereas distributed decisions are based on both product data and local events. The evaluation environment supports measurement of the operational performances of this decision architecture with regard to business and process disturbances and to a classical centralized approach.

3 Generic object-oriented components for emulation modeling

The proposed evaluation environment consists of an emulation-modeling framework, a prototype implementing this framework and an experimental protocol.

3.1 Modeling principle

The principle of an emulation-based experiment is to accept that proving $(SF_m \wedge CS \supset G)$ logically implies that $(SF \wedge CS \supset G)$, i.e. that a successful run of a control system on an emulated shop floor means that an application of this control system on the *real* system will also be successful.

Nevertheless, a model being an abstraction of reality, this implication can be questioned. For instance, it is a usual practice not to represent routing issues in simulation models: entities representing products are transparently moved to their programmed destinations. Such simulation models are simpler, but do not allow studying routing decisions.

Therefore, an emulation model must not only represent reality correctly, but also comprise all the *important* aspects of reality. The importance of an element is determined by its impact on the success of the experiment. In summary, *an emulation model must include what might cause a negative experimental result.*

Several key properties of the benchmarking environment can be deduced by applying this principle to product-driven control issues. Thus, as product-driven systems might provide a natural solution to synchronization issues between physical and informational flows, the benchmarking environment must by default *separate* the different kinds of flows (matter, data, events). Likewise, the emulation model must be neutral with respect to production execution functions: on one hand, every action physically possible can be executed (including routing); on the other hand, no action should be executed without an explicit request from the control system. Finally, as the product may have an active role in production execution, the modeling components must have sufficient resolution to treat each product individually.

3.2 Systemics-based modeling components

Our methodology for building an emulation model is based on a systemic vision of the shop-floor system. As we focus on products, we aim at representing their physical evolution. According to systemics, these evolutions can be modeled as shape, space and time transformations (Feliot, 1997).

The first step of the modeling methodology is a shop-floor analysis, in order to determine every physically possible product life cycle. A state-transition approach is used: states correspond to stable product positions and shapes, whereas transitions model physically possible spatial and morphological transformations.

In the second step, we aim at modeling actuation on products. Transitions between product states are implemented by shop-floor equipment. Therefore, we introduce two modeling constructs,*shape* and *space transformations*. These constructs must take into account physical constraints such as resource cycle and setup time, capacity limits, etc. Moreover, a third construct, *time transformation*, is defined, dedicated to modeling products waiting between transformations (Figure 2).

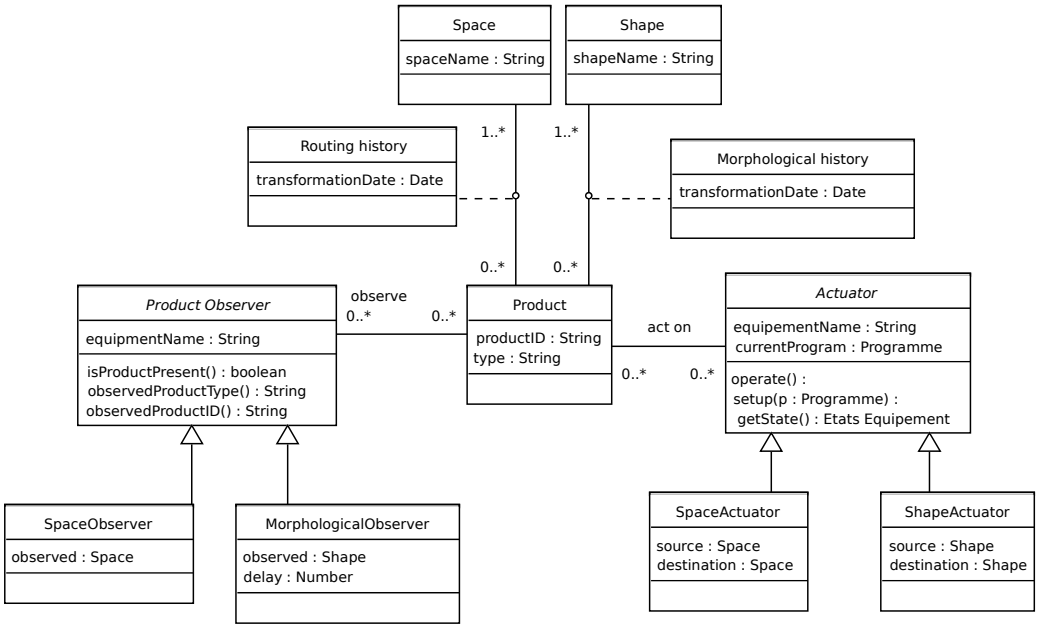

Fig. 2. UML class diagram: emulated products are characterized by their position (space attribute) and morphology (shape attribute).

Each shape and space actuation bloc offers an interface, which enables an external system to interactively control them. Control messages enable us to request the transformation to be set up or to begin operating. Conversely, report messages permit knowing the current state of the resource. Each shape or space actuation bloc is configured with a set of programs (Figure 3). Each program represents an operation that can be done by this actuator; it is modeled as a delay and a change in the product shape or space attribute. A probability distribution function may be used to specify this delay, as well as setup times and availability parameters (i.e. mean time to repair, mean time before failure).

Using these three kinds of modeling constructs, we were able to describe the shop-floor structure in a generic way.

Finally, the emulation model would not be complete without a way to observe products, so the third step of the methodology is concerned with product sensing. These modeling constructs include physical laws such as limits of scope and sensing accuracy. Three different accuracy levels have been defined:

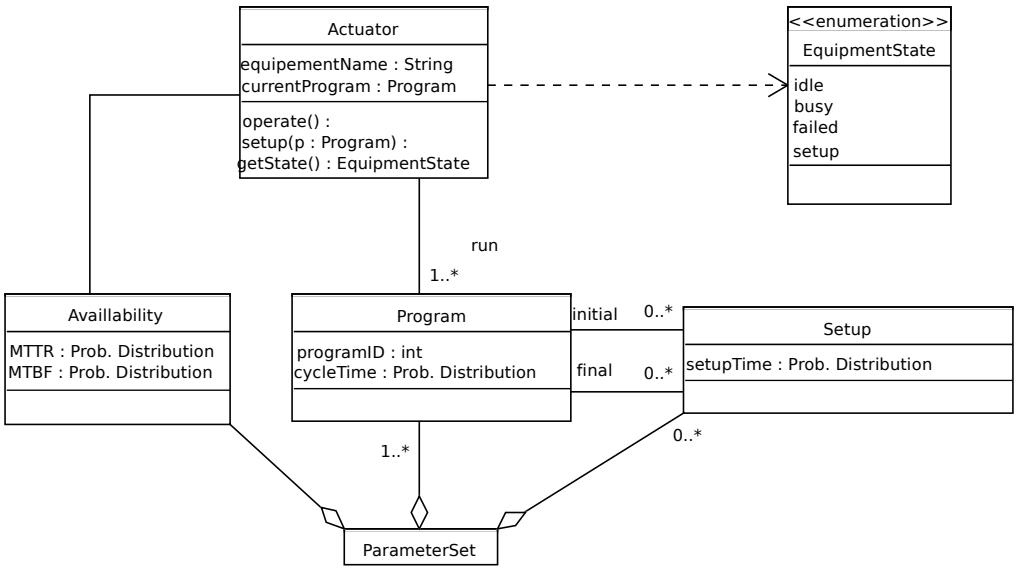

Fig. 3. UML class diagram: actuators parameters. Possible actions are represented as programs. Equipment availability and sequence-dependant setup can be modeled.

identification of an individual product (an RFID transponder is a concrete example of such sensing equipment), identification of the product type, or plain detection of a product presence.

By representing the physics of products, actuation and observation capabilities, we have modeled the lower half of a cybernetic loop that can be interfaced with a control system by exchanging request and report messages (Figure 4).

In conclusion, these modeling constructs enable us to build an emulation model from the point of view of logistics. They aim at modeling the physical laws constraining product actuation, transformations and sensing, while offering interfaces to actuators and sensors.

3.3 Implementation of a prototype

A prototype supporting the emulation methodology has been developed using Arena Professional. Each modeling construct is implemented as a template simulation module. Moreover, interaction capabilities were developed into an external library (DLL).

Interaction with emulation modules can be done either from a Visual Basic program embedded in an Arena model, using direct API calls, or from an external system, sending XML-encoded messages across a TCP socket.

On one hand, the API emulation enables embedding a simplified control sys-

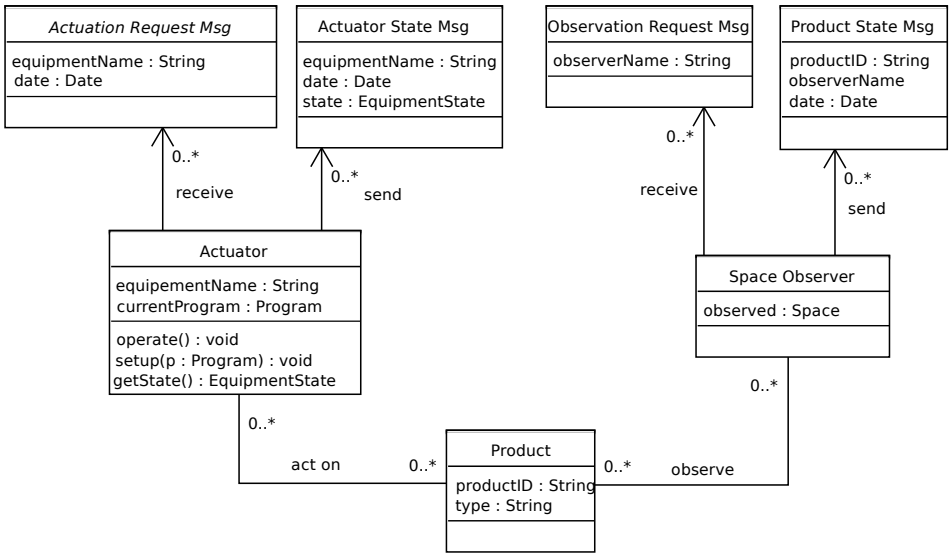

Fig. 4. UML class diagram: the three main component types are structured to form the lower half of a cybernetic loop.

tem directly into the simulation software. The control system must be implemented as a set of control functions and must be able to interact with external tools such as a database, or enterprise information system. From the point of view of the control designer, this mode enables quick and easy definition and testing of control algorithms. From a technical point of view, it allows conducting discrete-event experiments easily, because both control and emulated systems share the same event scheduler. Progressing time by scheduling discrete events makes it possible to run many experiments, and/or to experiment over a large time-span.

On the other hand, a remote control system can be connected to the emulation model by exchanging control requests and responses over the network. This interaction mode allows more in-depth analysis of the proposed control architecture, as the actual control software is used. Therefore, it enables performance evaluation of a particular implementation of the aforementioned control algorithms. Nevertheless, from a technical point of view, it is hard to use discrete events ahead of time because the control system cannot easily get access to the emulated system event-scheduler. While solutions have been proposed to solve this problem, such as hybrid discrete-event/real time emulation (Saint Germain et al., 2003) or synchronization based on HLA, it remains hard to solve. A workaround solution consists in using only the real-time mode, but this proves impractical if experimenting over a large time-span.

3.4 Validation

Validation of this prototype was done first using unitary testing by modeling simple cases to ensure that the emulation modules were correctly coded. These cases had known behavior that has been compared to emulation models behavior in order to ensure correct implementation of the emulation modules.

Then, the modeling methodology has been tested against several industrial case studies to validate its applicability. The first case is a furniture manufacturer, characterized by high product variety, a reference-specific bill of operation, and strictly FIFO work-in-process queues. The modeling components have proved able to represent all the major aspects of this case. Compared with older model of the same shop floor, this model enabled finer control of product flows, such as precise control of product routing. In a way, modeling was easier with the emulation component, as the modeling process was focused only on the physical properties of the shop-floor equipment, without having to take into account control issues (often very hard to include in classical simulation models (van der Zee, 2006)).

Nevertheless, this application has also put the emphasis on some of the limitations of our prototype, such as the inability to automatically set the parameters for the emulation modules, or the need for an improved error handling mechanism.

To conclude this section, we can state that the proposed modeling methodology helps designing emulation models of complex large-scale industrial systems, even if the implemented prototype may require further developments. These emulation models may be used to conduct product-driven control experiments. The next section presents such an experiment.

4 Application to an industrial case study

4.1 Case presentation and parameters

The case of an automotive-industry subcontractor was studied. The assembly site can produce up to 10,000 products a day, with product references in the hundreds. The factory is actually divided into several production cells, each including every production step to make a finished product from raw material, and is dedicated to a particular client. One of these cells has been modeled.

As shown figure 5 the production process is divided into two stages. A first

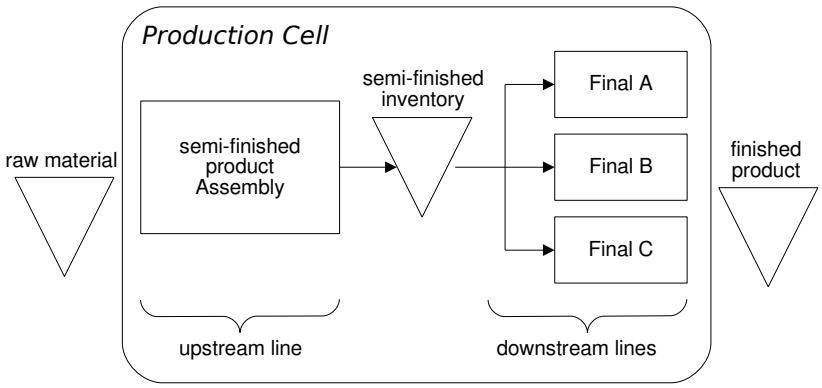

Fig. 5. Material flow in a production cell

set of operations on the first line (called SF) results in semi-finished products, which are further assembled on three independent assembly cells (called FA, FB and FC, or collectively F). We assume that all assembly cells cannot work simultaneously on the same reference. The production module therefore includes, in addition to the four cells, an inventory storing semi-finished products.

As downstream F lines can consume three kinds of products may simultaneously, the major decision problem is to choose whether and when to change the setup on the upstream line SF. This decision aims at minimizing the number of setups, while also minimizing work-in-process levels and avoiding starvation on downstream lines. Moreover, business goals such as due dates must also be taken into account.

We modeled this shop floor using shape transformations for each of the four cells. Stores were modeled as time transformations, whereas two space transformations were used to move semi-finished products in or out of the store. Finally, some product sensors have been added to the model to trace emulated products.

The emulation modules have been configured according to the cycle times given in Table 1. Furthermore, the setup time has been configured according to a normal probability distribution (with a mean of 20 min, and a standard deviation of 10 min), while transport delays have been fixed to 30 s. These data correspond to measurements made on the real system and to the goal of the experiments.

4.2 Decision-making procedures

As stated in the problem statement section, the goal of this experimental study was to compare product-driven control algorithms with a traditional

product reference	cycle time (SF)	cycle time (F)
1	1.34	3.6
2	1.06	3.49
3	1.14	3.7
4	1.14	3.57
5	1.17	3.54
6	1.13	3.75
7	1.22	4.09
8	1.22	3.69
9	1.11	4.03
10	1.23	3.62
11	1.23	3.75
12	1.11	3.96

Table 1
Cycle times used in the emulation model (minutes)

centralized approach, and to compare various product-driven control algorithms. Therefore, a centralized control algorithm and two different product-driven control algorithms have been defined.

The centralized reference control system schedules jobs according to their critical ratio, defined as the ratio between the processing time required to complete it and the time available before its due date. This predictive schedule is then implemented by decision centers, with jobs being delayed in case of process disturbances. Jobs are rescheduled when business disturbances happen. This classical algorithm serves here as a comparison point.

To model product-driven decisions, every product is represented as an object, its attributes representing product-embedded data. A component is dedicated to synchronizing products with their physical counterparts and to enable decision centers to access products.

One of the main product attributes is priority. At system initialization, the central scheduling procedure is run, and product priorities are assigned according to the schedule. Other attributes of products are their reference, used to setup shape transformations, and their state, enabling advancement of products through their bill of operations.

We have programmed two modes of product-driven control. In the simpler one, local control decisions are based on products priority *only*: When a cell is ready to operate, it scans through the products waiting and selects the

one with the highest priority. This control algorithm is called "product-driven constrained" because it depends on the centrally made schedule.

In the other one, a more complicated decision algorithm is used, allowing autonomy with respect to product data. This algorithm tries to mimic the behavior of an operator, taking into account not only product priorities, but also other local parameters, such as levels of semi-finished inventory in order to avoid starvation downstream, or the amount of products of the same reference that have been done, to minimize setups. This control algorithm is called "product-driven autonomous".

4.3 Experimental protocol

The experiment design chosen to conduct the experiments enabled us to consider every important combination of factors methodologically, to measure the effect of each factor, and finally to study the system sensibility to each factor.

4.3.1 Experimental scenario

After the physical system was analyzed, modeled, and configured with process cycle and setup times, a production scenario was specified in the form of a set of production orders (Table 2).

product type	quantity to make	due date
1	1775	7200
2	192	3600
3	15	3600
4	832	7200
5	448	7200
6	18	7200
7	600	7200
8	192	7200
9	240	7200
10	640	7200
11	128	7200

Table 2
Production scenario

Factor	Modality	Comment
Process disturbance (PD)	0	no resource failure
	1	short and frequent failures
	2	rare and longer failures
Business disturbance (BD)	0	no order modification
	1	rush order
	2	quantity modification
Control mode (Ctrl)	0	reference centralised control
	1	product-driven control with autonomy
	2	product-driven control without autonomy

Table 3
Experimental factors and their modalities.

4.3.2 Factors

We can define three generic factor categories:

- The *control mode* is obviously one of the more important. A reference control system must be included, to provide reference performance measures.
- *Process disturbances* pertain to the various disrupting events on physical operations (for instance, resource failure or product scrapping).
- *Business disturbances* correspond to variation on demand (including rush orders as well as variation in quantity or priority of existing orders).

In the current experiment, three modalities have been defined for process disturbances (PD), focusing on resource availabilities. There is either no failure, or shorts and frequent failures (uptime = expo (600), downtime = norm (45; 6)), or rare and longer failures (uptime = expo (3000), downtime = norm (360, 180)). These availability data have been inspired by maintenance logs of the real system.

Business disturbances (BD) are rush orders, involving small quantities (25) with a very short delay (1.5 times the production time), and order modifications, where the quantity is larger (125), but the delay is longer. The last factor is the control mode (Ctrl), which can be either centralized, constrained product-driven or product-driven with autonomy. Table 3 summarizes the various factors and their modalities.

4.3.3 Performance indicators

According to the goals of the experiment, we focused mainly on production performance in order to evaluate the impact of product-driven control on productivity gains. Several classical performance indicators may be considered: *Overall equipment effectiveness* (OEE), *lead time* (LT), *work in process level* (WIP), and *tardiness*.

In the current experiment, WIP and LT were used. OEE is highly correlated with LT and is therefore not shown in the graphs.

Moreover, using a specifically developed data-processing tool, each experiment can be presented as a Gantt chart, enabling a finer analysis of the experimental results.

4.3.4 Experiment design

To study the effect of the factors on the performance indicators, experiment design was used. A complete combinatorial design based on a L_{27} Tagushi table was used to conduct the experiment.

For every experiment in the design, 15 replications were made, to ensure statistically correct results. This number of replications was chosen empirically by considering confidence intervals. This resulted in more than 400 replications that take about 30 h to complete on a desktop-class computer.

5 Experimental results and discussion

5.1 Validation of control algorithms implementation

The first step before conducting the actual experiment design is to make sure that the control algorithms have been correctly defined and implemented. Indeed, a development error may introduce a serious bias in the results.

To achieve this verification, we compare each control algorithm in a nonperturbed situation. Gantt charts for each control model are presented in Figure 6 and in Table 4. In these charts, we can see that the constrained product-driven algorithm mimics the centralized algorithm with high accuracy. Therefore, in a nonperturbed situation, product priority enables us to find the global schedule back.

The autonomous product-driven control behaves slightly differently, but yields

similar operational performance values.

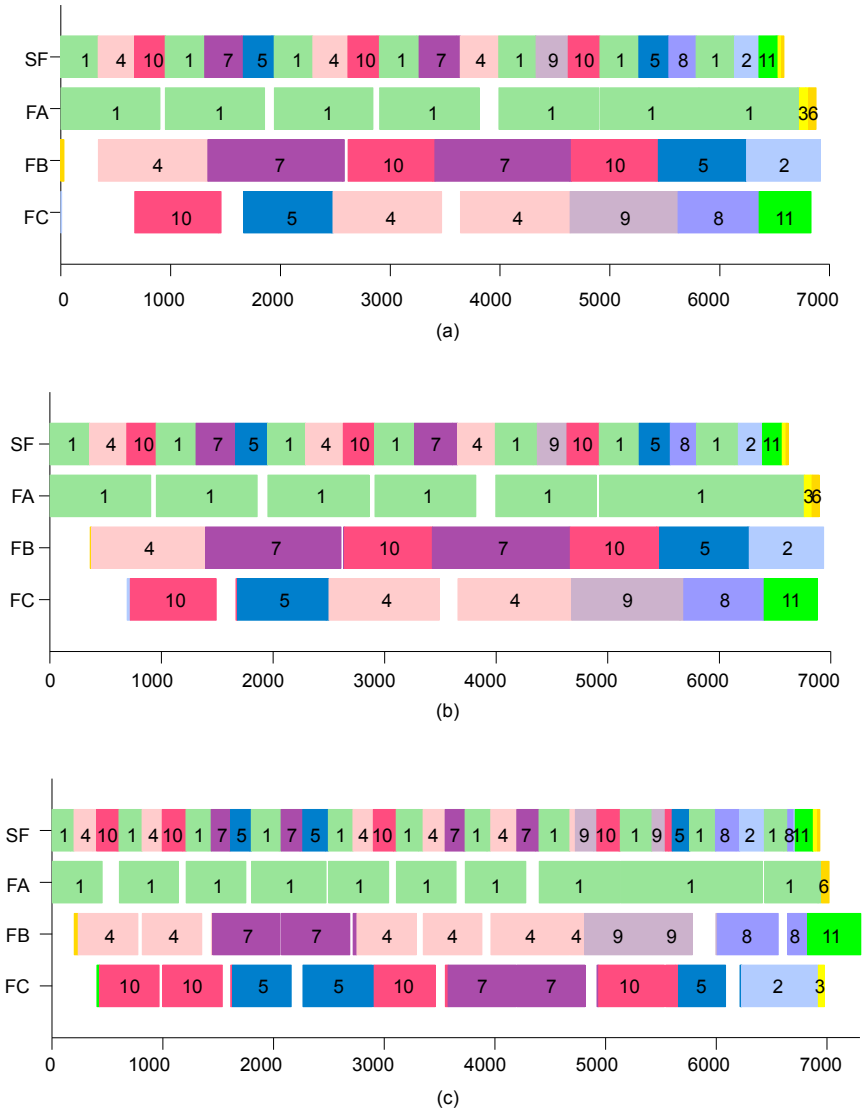

Fig. 6. Gantt chart representing experimental results for centralized control (a), product-driven control without autonomy (b), and with autonomy (c). SF, FA, FB and FC are the name of the manufacturing lines. The abscissa of each graph is time (in minutes).

5.2 Effects

The following graphs show the effect of each of the factors on performance indicators (Figure 7). The first three curves are plots for process disturbances, business disturbance and control mode, respectively; the last two are the effect of interaction between process disturbance and control and between business

control modality	lead time	WIP	overall effectiveness
centralised	6985,57	250,78	88,70%
product-driven autonomous	6995,53	260,55	88,59%
product-driven constrained	7185,76	141,29	86,23%

Table 4
Performance indicators values for each control mode in a non-perturbed situation

disturbance and control.

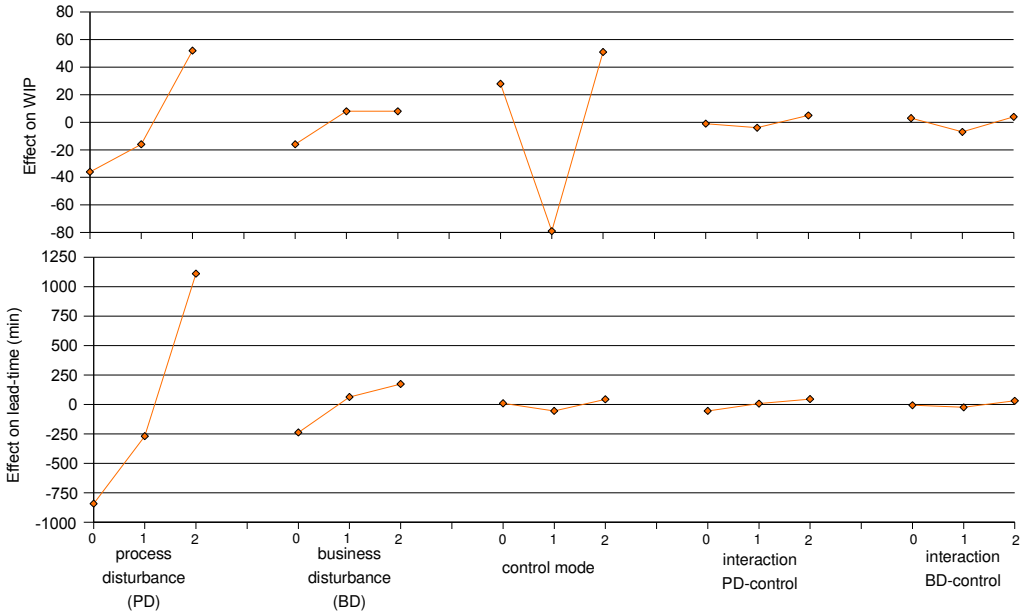

Fig. 7. Effect of various factors on work-in-process level and on lead-time

According to the experimental results, product-driven control (with autonomy) causes lower WIP levels and a slightly lower lead-time than centralized control. Conversely, product-driven control without autonomy shows worse performances than centralized control.

The last two curves show that there is an interaction, but a more precise analysis would require more investigation. The following graphs enable easier data analysis. They show on the same graph the effect on WIP (Figure 8) or on LT (Figure 9) of the interaction between process disturbance (PD-Ctrl) and control mode and of the interaction between business disturbance and control mode (BD-Ctrl).

The first graph shows that the effects of disturbing factors dominate the effect of control mode. Nevertheless, in case of high perturbation (points 2, 5 and 8), autonomous product-driven control is the best. Likewise, this control mode is the best for some business disturbances (rush orders).

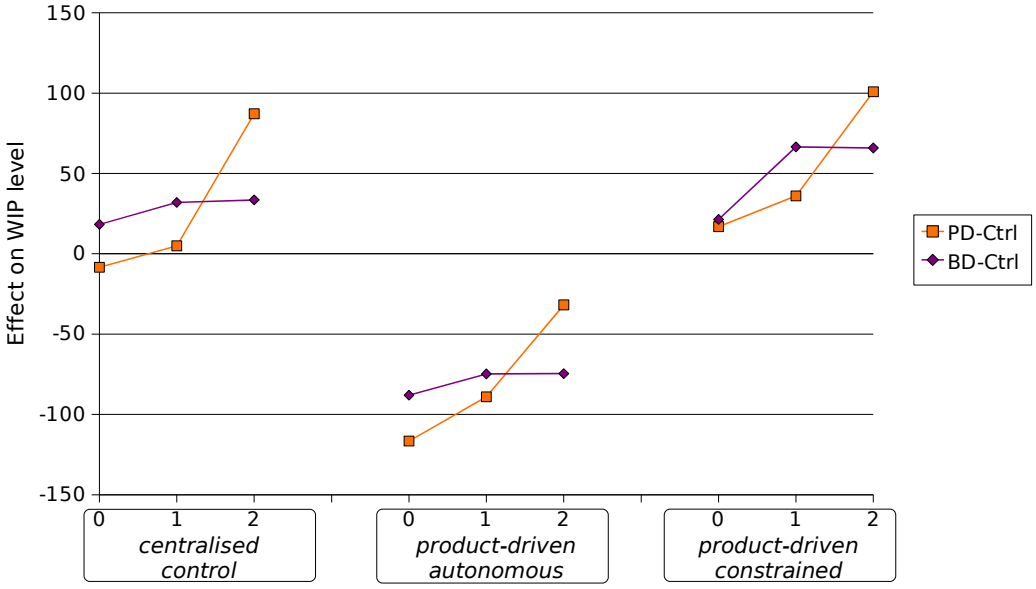

Fig. 8. Effects on work-in-process levels of interactions between control mode and disturbance factors

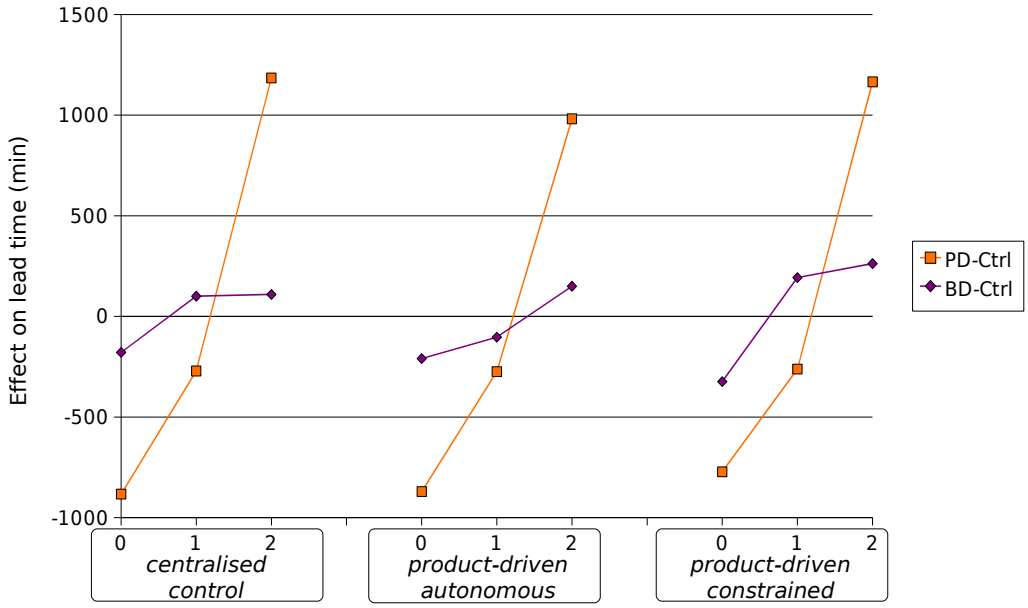

Fig. 9. Effects on work-in-process levels of interactions between control mode and disturbance factors

The second graph shows that the control mode has a clear impact on WIP levels. Indeed, autonomous product-driven control proves to be the best for every disturbance configuration.

5.3 Discussion

These results show that a product-driven control approach enables combining the good performances of centralized control in nominal situations with the adaptability and robustness of distributed control.

It is noteworthy that these experiments also demonstrated that local decision-making and its accuracy are crucial in a product-driven environment. Indeed, without autonomous decisions, performance is similar to that of centralized control, when there is no perturbation, but performance tends to plunge in a highly perturbed environment. In contrast, with autonomous decisions, product-driven control was slightly less successful in non-perturbed situations, but showed more robustness.

Another point that should be discussed is the level of "technical intelligence" of local decision-making algorithms. Indeed, we have used simple decision procedures that sort products using a simple or more complex function. These functions were inspired by the behavior of operators in the shop floor. As we have seen, performance of product-driven control depends highly on the nature of the local decisions. In a real industrial application, where more advanced artificial intelligence or real human (intelligent) operators make decisions, a product-driven control might show better performances. Indeed, a human operator may be able to take into account more operating parameters than our artificial decision rules do. Moreover, a human might be able to adapt to changing operational conditions. Finally, by using more intelligent actors, it would be possible to know if the system operates in normal or degraded condition, and to change the balance between predicted centralized decisions and opportunistic local decisions accordingly.

6 Conclusions

A modeling methodology based on generic systemics-inspired building blocks has been presented, as well as a prototype allowing an incremental experimental approach. The modeling approach, based on the separation between an executable model of the shop floor system and the control system, enabled us to setup an experimental design where several control systems are compared. Moreover, the modeling constructs introduced enabled to simplify the

modeling process: by providing an abstract view of the technical equipments to model, it is possible to focus on the most important point, the logistical properties of the shop floor. This modeling approach has been validated by its application on an industrial case-study.

Moreover, interesting properties of product-driven control have been highlighted by this application. The first one is the demonstration that the performance of a product-driven decision system can be as high as the one of a traditional system. Indeed, in a non-perturbed situation, centralized decisions such as a schedule can be translated into products attributes, allowing to locally reconstructing the centralized decision without notable performance loss.

The second main finding of the benchmarking case-study is that the robustness of a product-driven system depends mainly on local decision procedures. Indeed, two different product-driven decision processes have been compared in various disturbance levels. On the one hand, taking into account only product attributes in the local decision process results in poor performance while facing disturbances. On the other hand, using both product attributes and local information such as resource state, inventory levels or current operating conditions enabled to achieve better robustness. Autonomy and "intelligence" of local decision-making seems therefore to be very important in a product-driven control approach.

Further experiments are nevertheless required to access the performance of product-driven control, taking into account software-related issues such as data exchanges between products, decision centers and physical equipments.

Moreover, the modeling methodology must be further developed and further tested for validation. Today, two other research works are using it (El Haouzi et al., 2008) (Klein and Thomas, 2006). The new applications will help extend our methodology toward more generality and stability. Its implementation must be continued to improve usability, eventually resulting in the release of a standalone tool.

Perspectives include using this evaluation environment to model more cases, to study scientific aspects of product-driven control, and to help in transferring product-driven control technologies to industrial partners, in the domain of natural fibers. Two industrial applications are in process: the first one in a furniture factory, the other in an air conditioning appliances company. For these transfers, emulation-based evaluation will be backed up by a physical test bed to take into account more technological problems, such as reliability of identification technologies or networking issues.

References

Babiceanu, R., Chen, F., 2006. Development and applications of holonic manufacturing systems: A survey. *Journal of Intelligent Manufacturing* **17** (1), 111–131.

Baina, S., Morel, G., 2006. Product centric holons for synchronisation and interoperability in manufacturing environments. In: *12th IFAC Symposium on Information Control Problems in Manufacturing INCOM'2006*.

Brennan, R. W., 2000. Performance comparison and analysis of reactive and planning-based control architectures for manufacturing. *Robotics and Computer-Integrated Manufacturing* **16** (2-3), 191–200.

Cavalieri, S., Garetti, M., Macchi, M., Taisch, M., 2000. An experimental benchmarking of two multi-agent architectures for production scheduling and control. *Computers in Industry* **43** (2), 139–152.

Cavalieri, S., Macchi, M., Valckenaers, P., 2003. Benchmarking the performance of manufacturing control system: design principles for a web based simulated testbed. *Journal of Intelligent Manufacturing* **14** (1), 43–58.

Corning, P. A., 2002. The re-emergence of "emergence": A venerable concept in search of a theory *Complexity*, **7**, pp. 18-30.

El Haouzi H., Thomas A., Pétin J.-F., 2008. Contribution to reusability and modularity of Manufacturing Systems Simulation Models: application to distributed control simulation within DFT context. *International Journal of Production Economics* **112**(1), pp. 48–61.

Fusaoka, A., Seki, H., Takahashi, K., 1983. A description and reasoning of plant controllers in temporal logic. In: *International Joint Conference on Artificial Intelligence*. pp. 405–408.

Kärkkäinen, M., Ala-Risku T., Främing K., 2003. The product-centric approach: a solution to supply network information management problem?, *Computer in Industry*, **52**(2). pp. 147-159.

Klein, T, Thomas, A., 2006. A simulation testbed for decision system evaluation in a furniture manufacturing group, 12th IFAC Symposium on Information Control Problems in Manufacturing, INCOM'2006, St-Etienne, France, pp. 573-579.

Marik, V., Lazansky, J., 2007. Industrial applications of agent technologies. *Control Engineering Practice.* **15**(11), 1364–1380.

Mcfarlane, D., Sarma, S., Chirn, J. L., Wong, C. Y., Ashton, K., 2003. Auto id systems and intelligent manufacturing control. *Engineering Applications of Artificial Intelligence* **16** (4), 365–376.

Monch, L., 2007. Simulation-based benchmarking of production control schemes for complex manufacturing systems. *Control Engineering Practice* **15**(11), pp. 1381–1393.

Monostori, L. and B. Cs. Csáji (2007). Production structures as complex adaptive systems. In: Proceedings of the 40th CIRP International Seminar on Manufacturing Systems, May 30 - June 1, Liverpool, United Kingdom. p. (CD version is available).

Morel, G., Valckenaers, P., Faure, J.-M., Pereira, C. E., Dietrich, C., 2007. *Manufacturing plan control challenges and issues* **15**(11), pp. 1321–1331.

Pannequin, R., Thomas, A., 2004. Centralized versus distributed decision, an industrial case. In: *11th IFAC Symposium on Information and Control Problem in Manufacturing INCOM'2004*.

Feliot C., 1997. Complex Systems Modelling: integration and formalization of models. PhD thesis, University in Sciences and Technologies of Lille.

Pfeiffer, A., Kádár, B., Monostori, L., 2003. Evaluating and improving production control systems by using emulation. In: *Applied Simulation and Modelling*.

Plossl, G. W., 1985. *Production and Inventory Control: Principles and Techniques* (2nd Edition). Prentice Hall, Englewood Cliffs, NJ.

Saint Germain, B., Valckenaers, P., Zamfirescu, C., Bochmann, O., van Brussel, H., 2003. Benchmarking of manufacturing control systems in simulation. In: *Proc. of the 3rd Int'l Workshop on Performance Measurement* (IflP WG5.7), Bergamo. pp. 357–369.

Valckenaers, P., 2001. Editorial of the special issue on holonic manufacturing systems. *Computers in Industry* **46** (3), 233–234.

Valckenaers, P., Cavalieri, S., Germain, B., Verstraete, P., Hadeli, Bandinelli, R., Terzi, S., Brussel, H., 2006. A benchmarking service for the manufacturing control research community. *Journal of Intelligent Manufacturing* **17** (6), 667–679.

Valckenaers, P., van Brussel, H., Bongaerts, L., Wyns, J., 1997. Holonic manufacturing systems. *Integr. Comput.-Aided Eng.* **4** (3), 191–201.

van der Zee, D. J., 2006. Modeling decision making and control in manufacturing simulation. *International Journal of Production Economics* **100** (1),

155–167.

Vollmann, T. E., Berry, W. L., Whybark, C. D., 1997. *Manufacturing Planning and Control Systems*. McGraw-Hill.