An Agent-Based Personalized Producer/Consumer Scenario

Christian Seitz[1] and Bernhard Bauer[2*]

[1] Siemens AG, Corporate Technology, Information and Communications
81730 Munich, Germany
Christian.Seitz@mchp.siemens.de
[2] Institute of Computer Science, University of Augsburg
86150 Augsburg, Germany
Bernhard.Bauer@informatik.uni-augsburg.de

Abstract. We present an agent-based Producer/Consumer scenario, that is the key element of manufacturing systems and therefore of utmost importance. We call it a MSWS-system, which is made up of a series of machines, shuttles, a warehouse and again shuttles. We applied a hybrid approach, i. e. which is a combination of a completely centralized and a totally decentralized approach. A controller supervises two MSWS-Systems and intervenes only in exceptional cases. On account of an increasing significance of the human factor in manufacturing systems, each machine operator is endowed with a user profile.

1 Introduction

The applicability of agents in the area of manufacturing has been studied very intensely in the last years. In this paper we focus on a small, but very important part of manufacturing systems, which we call a MSWS-System, a series of *M*achines, *S*huttles, a *W*arehouse and again *S*huttles. Such a system appears in any manufacturing system and often a MSWS-System is directly followed by another one. Since the productivity of people is in relationship with their motivation, we endow each operator and machine with a profile, which enables the production system to integrate the workers wishes in the manufacturing process. In order to guarantee, that all incoming orders are completed in time, each component is endowed with a load balancing mechanism. Coalitions of machines are formed to complete an order, with the consequence that a failure of a single machine harms less, because there is always a backup machine present.

Agent based manufacturing systems have been studied intensely in the literature in the last years. Design issues on agents and manufacturing system are illustrated in Bussmann *et al.* [?]. Shen and Norrie [?, ?] give a detailed overview of developed systems and mechanisms. Architectures of manufacturing systems

---

[*] Former address: Siemens AG, Corporate Technology, Information and Communications.

Fig. 1. The MSWS-Scenario in Manufacturing Systems

are shown by Usher and Wang [**?**]. They differentiate between a hierarchical approach, where machines are supervised by a hierarchy of controllers and a *heterarchical* approach with a flat organization. We follow a hierarchical approach, but we only introduce one controller-layer. Manufacturing systems, which give the workers more competencies [**?**] are called human centered systems (HCS). In our system each worker has a user profile, which enables him to influence the system. There are several possibilities to control agent based manufacturing systems. Kis *et al.* [**?**] use a market mechanism and Chen *et al.* [**?**] apply negotiations and Bussmann and Schild [**?**] use auctions. We use a the contract-net for negotiation among the agents. Our primary goal is to conflate manufacturing systems and personalization.

The paper is structured as follows. In Section 2 we define our work. Section 3 gives an overview of the architecture of our system. Section 4 introduces the planning system. In section 5 we present some optimization aspects and section 6 concludes the paper with a summary.

2 Problem Definition

This section introduces the Producer/Consumer scenario. Furthermore, organizational aspects of manufacturing system are explained. The section concludes with some assumptions about the system.

2.1 Shop Floor Scenario

We focus on a small but very important part of manufacturing systems, which appears in every system. It is a Machine-Shuttle-Warehouse-Shuttle-(MSWS)-System (see figure 1A). A machine produces goods, which are transported by a shuttle[1] to a warehouse. If the products of the warehouse are needed for further production they are brought to another machine by a shuttle. The number of machines a M-subsystems consists of is not limited, but they all belong to the same machine type, e. g. a cutter or a CNC lathe. The same applies to the shuttle- and warehouse-subsystem. This small scenario is omnipresent in any shop floor and often a MSWS-System is followed by another one as depicted in figure 1B.

2.2 Preferences and Profiles of Humans and Machines

In manufacturing systems the humans working at the machines are often neglected. They do not have any possibility to intervene in the system. However,

---

[1] A shuttle is a kind of vehicle, which navigates autonomously in a shop floor.
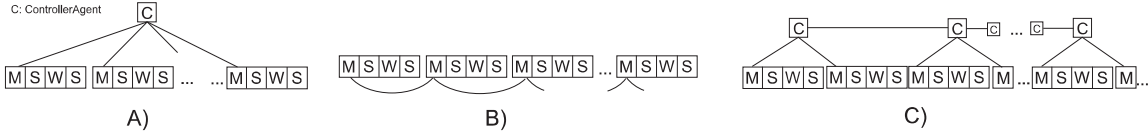
Fig. 2. Different organizational approaches

there is a tendency that the significance of humans in the production process will increase (see e.g. Choi and Kim [**?**] or Barbuceanu and Fox [**?**]). Giving a person more competencies, people are more motivated and this directly induces better work. Experience shows, that people defining their rest periods by themselves work faster after the break and make less errors. Therefore, we equip our manufacturing system with personalization aspects. Each worker is characterized by specific profiles for particular machines, like clock time, machine settings etc. Beyond machine specific profiles of a human, his preferences are taken into account, like break times or preferred working hours. Additionally, we associate a profile with each machine. The profile of a machine contains e.g. information about downtime, varying cycle times or production times. There are constraints regarding the entries in a machines' profile. An operator may not enter a cycle time in the machines profile with the consequence that the orders currently processed cannot be completed in time.

2.3 Organizations

There are three different approaches for the architecture of an agent-based manufacturing system. The first one is a totally centralized approach, depicted in figure 2A. A central controller agent (C) supervises all machines in the manufacturing system. This component is able to synchronize and coordinate all its subsystems and is able to perform a global optimization. But a breakdown of the central component disrupts the production and it is not easy or even possible to integrate rush orders in a running system, because replanning is too time consuming.

The opposite is a completely decentralized approach (see figure 2B). Each machine possesses its own local planner and performs local optimizations based on its local view of the world. The local planners communicate with other local planners of previous or successive machines. Thus, a machine can respond faster to local changes of the environment, since no coordinator is involved. However, if the system is complex and a problem occurs at the end of the assembly, the machines at the beginning do not know anything about the trouble. The longer the assembly line, the more time will elapse until information gets from one end of the line to the other.

The hybrid approach (see figure 2C), which we follow for the rest of the paper, combines the advantages and eliminates the disadvantages of the two previous mentioned methods. There are multiple coordinators, each responsible for a fixed number of machines. The coordinators communicate with each other to

exchange information about their machines' behavior. The controllers enforce the communication between the machines and the local planners preserve flexibility. A breakdown of a single controller is less crucial in a hybrid system than in systems with only one coordinator.

2.4 Definitions and Assumptions

Before we describe our system in detail, some definitions and assumption are made. An order o consists of a product-id (pid), which specifies the product being made by a machine; the number of goods n which have to be produced, and a deadline t_d which specifies the time, the order must be completed. A *coalition* of machines is completing an order. A warehouse is made up of a fixed number of storage places, each capable of taking up a fixed number of goods. In a storage place only one type of product (same pid) is stored. We emanate from an shuttle surplus and we assume that shuttles will never break down. Finally, a profile of a machine is defined as follows:

Definition 1. *Let* M *be a machine. A profile **Prof(M)** of a machine* M *consists of a set of machine properties* M_p, *such that for each resource[2] type* m_i, *a set of PropertyNames* N_{Prop}^i *exist. The PropertyValues* V_{Prop} *are always (for each resource type) non-negative integers. A MachineProperty* M_p *of a machine* i *is a pair* $N_p^i \times V_p$.

Preferences PREF(U) of a user U are modelled as described in Kießling [**?**] and are a set of preference values V_{Prop} and preference attributes A_{Pref}^i. A relation $x \preccurlyeq y \subseteq V_{Prop} \times V_{Prop}$ is interpreted as, *a person likes y better than x*. With this, a user profile UP can be defined, using PROF(M) and PREF(U).

Definition 2. *Let* $\{PROF(M_i)\}$ *be a set of machine profiles* M_i *and* U *the users' preferences* $PREF(U)$, *a user profile* $UP = \{\{PROF(M_i)\}, PREF(U)\}$.

A workers' profile consists of one or more machine profiles[3] plus additional information about the users' preferences, e. g. duration of coffee breaks.

3 Architecture

In this section the architecture of the MSWS-System is described. It consists of a M-, S-, W-, and another S-subsystem. Each subsystem is made up of a fixed number of resources of the same type and a single machine is controlled by several agents, called the *Agent based Processing Unit*. Due to the hybrid approach, an additional control-layer, called the *Agent based Controller Unit*, is needed.
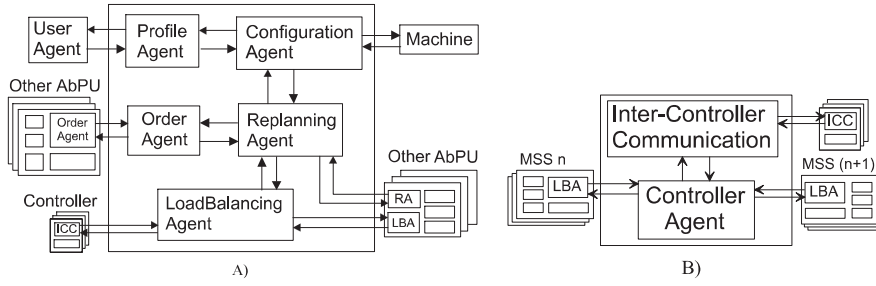
3.1 The Agent Based Processing Unit

The Agent based Processing Unit (AbPU), envisaged in figure 3A, consists of six agents. The *OrderAgent* OA is in charge of sending and receiving orders from

---

[2] We call a warehouse, a machine, and a shuttle resources for unification reasons.

[3] A worker does not always work with the same machine type and therefore for each type a profile is needed.

Fig. 3. Architecture of the Agent based Processing Unit (A) and the Agent based Controller Unit (B)

other subsystems. The received orders are administered by the OA in a priority queue. With the *ProfileAgent* (PA) the machine operator is able to change the profile of a machine and to transfer its own profile, residing on its UserAgent, to the machine. The *LoadbalancingAgent* (LBA) checks regularly the order queue to get information about the workload of the machine. This data is forwarded to all other LBAs. The *ReplanningAgent* (RA) is always informed when an exception occurs, e. g. a machine breaks down. The RA arranges the orders to make sure the order with the highest priority is served first. The *ConfigurationAgent* (CA) is informed when either the profile has changed or the machine breaks down. With the *UserAgent* a worker is able to edit his or the machines' profile and can transfer the profile to the PA. A profile is specified in XML according to the definition given above. The PA sends the changes of the profile to the RA. This agent tries to customize the machine. Only if all changes can be realized, the new profile is accepted, else it is declined.

The architecture of the shuttle- and warehouse-subsystem differs slightly. They do not contain a RA because it is not possible to transfer products from one storing place or shuttle to an other. However, the LBA is still present, but there is only a communication channel *to* the controller unit to forward availability bottlenecks.

3.2 The Agent Based Controller Unit

The agent based controller unit (AbCU) consists of two agents, the Inter-Controller-Communication-Agent (ICC) and the Controller Agent (CA) (see figure 3B). A AbCU is responsible for two MSWS-systems (see figure 2C) and it sends instructions only to the machine-subsystems. If a LBA detects, that all machines in a subsystem are too heavy loaded, the CA is informed by a LBA. The ICC communicates with all other AbCU in order to get information about the load situation in other subsystem. If necessary, the ICC informs the CA. The CA informs controllers of preliminary MSWS-Systems, in order to increase or decrease production, and informs successive controllers about the future developments of the orders. The coordinator intervenes only in exceptional cases, with the consequence that a breakdown of a AbCU is less crucial in hybrid system than in systems with only one controller.

3.3 Communication

Figure 4 depicts the messages sent by each subsystem. An order has its origin either from an ERP (enterprise resource planning) or from an previous MSWS-System. At first a coalition of machines must be found, able to complete the order in time. Either the order is released by a machine, or an order is requested by the OA of a single idle machine. In the former case, the subsystem releasing the order contacts all machines in a M-Subsystem. Each AbPU responds with the percentage of the order it is able to complete in the specified time. The machine, which is able to produce most of the pieces gets the complete order and is called m_w, the others m_{l_i}. If there is a draw, the machine, which has answered at first gets the acceptance of the bid. Now, at least one *collaborator* is needed. m_w initiates an *iterated contract net* protocol and invites all other machines to participate in producing the order. Each m_{l_i} returns the number it is willing to produce. The m_w evaluates the incoming messages and depending on the order size and due date, m_w chooses an appropriate (see section 5.2) number of supporting machines, which are finally informed by m_w. If the number of pieces produced by the other machines is insufficient, another iteration of the contract net must be started. In the first run the m_{l_i} only return the number of pieces they are able to produce without replanning. In the second run the m_{l_i} reorganize the queues of each OA. If even this is not successful the controller is informed, which starts a *Reorganization mechanism* (see section 4.2). Until the order is processed the LBA can change the number of pieces each coalition member has to produce. In the latter case - a single machine requests a new order on account of outrunning orders in the waiting queue - the machine gets the order and now the aforementioned negotiation-procedure is started.

If a shuttle is needed to transport a workpiece to the warehouse, the machine sends a request message to the shuttle pool. Because of an unlimited number of shuttles, the request will always be successful. The shuttle, transporting a product, has to find a storage place, where workpieces of the same product are stored. The shuttle contacts all storage places and if there is a storages place containing the *pid*, and it is not totally filled, the workpiece is transported to this storage place, otherwise an empty one must be used. With each workpiece in the warehouse a *resting counter* is associated. Every time a new product arrives in
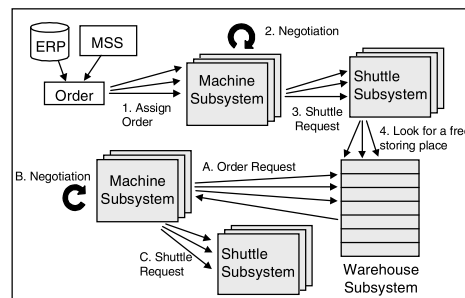
Fig. 4. Communication in a MSWS-System

the warehouse the resting counter for each piece is increased. This enables the successive MSWS-system to identify the workpiece staying the longest time in the warehouse. If the number of free storage places falls below a threshold, the detecting LBA informs the controller, which is now responsible for either reducing the output of the producing machines or starting only orders, containing *pids*, already present in the warehouse. But more frequently, a machine of a successive MSWS-Systems sends a request to all storage places of the warehouse. Either the most urgent workpiece (deadline) or the workpiece spent the longest time is chosen. Now, the machine is able to continue its work and a complete MSWS-System-cycle is passed.

4 Exception Handling

In a manufacturing system exceptions may occur at any time. We differ between *local* and *global* errors. Local errors only affect one MSWS-System, then a *replanning* is started. In global errors at least two MSWS-Systems are involved, and a *reorganization* is initiated. A replanning is tried first and only if this action is without effect, a reorganization is initiated.

4.1 Replanning

The LBAs periodically examine the input-queues of their machines and if not all orders can be fulfilled or a machine is broken down, the RA is informed and a replanning process is started. Breakdowns of machines conform to a negative exponential distribution ($f(x) = \lambda e^{-\lambda x}$ with the mean value $\mu = \frac{1}{\lambda}$). According to statistical estimation theory a $(1 - \alpha)$-confidence interval is $\left[\frac{2n\bar{X}}{c_1}, \frac{2n\bar{X}}{c_2}\right]$. $\bar{X}$ is the average of all downtimes and n the number of breakdowns. The two constants c_1, c_2 are the $\frac{\alpha}{2}$ and $1 - \frac{\alpha}{2}$ values of a χ^2 distribution with $2n$ degrees of freedom. If a machine breaks down, we first estimate how long this might be. In case, the estimation results in a very short downtime, the replanning process needs not to be accomplished, because in spite of the incident all orders may be completed in time. The situation changes, when the estimated downtime exceeds an upper threshold and finishing an order early enough becomes impossible. In this case, the coalition members take over a little amount of extra work. However, it may happen, that the support of all coalition members is still insufficient. In such situations, the number of products, the broken machine ought to have made, must be transferred to another machine. This has the same effect as adding a completely new order to the system. In case of a breakdown there are four possible activities which might to be done:

1. No action has to be performed, if the estimated downtime predicts only a short-time failure.
2. The downtime is rated as too long, in this case parts of the order must be transferred to coalition members.
3. If all coalition members are unable to complete the order in time, a replanning is initiated.

4. If all aforementioned actions fail, a reorganization is started by the AbCU.

If the machine is reconfigured by changing its or the operators profile, the estimation step is omitted and only the two subsequent steps are executed.

4.2 Reorganization

If the behavior of two or more MSWS-Systems escalate, a reorganization procedure is started by the AbCU. There are two situations, when a reorganization is initiated. Either a preceding replanning was not successful or the LBA of any subsystem detects an error, from which a MSWS-System cannot recover by itself. The AbCU is always informed by the LBA about a machines load, properties and state. According to the received messages from the LBA, the AbCU may carry out a set of actions:

- Initiation of a replanning in other MSWS-systems.
- Increase or reduction of the output of an arbitrary MSWS-system.
- Information of MSWS-systems about orders in advance (e. g. rush-orders).

5 Optimization

This section describes how load balancing is achieved and how the optimal size of a coalition for an order is computed.

5.1 Load Balancing

Load balancing algorithms try to distribute the load equally among all system components [**?**]. In the AbCU the LBA, is in charge of distributing the load among all members of the subsystem. The LBA permanently observes the queues of the OA and decides whether orders are sent to other OAs.
At first, we calculate the time t_o, that is needed to complete the order o. It has to be taken into account, that a machine must be rebuild or may break down. These intervals must be additionally considered when calculating t_o.

$$t_o = t_{rb} - nt_p \cdot (1 + p_f t_{rep}) \tag{1}$$

t_o is the estimated time a machine needs to produce n pieces of an order o. p_f is the probability for the machine to break down and is defined as $p_f = \frac{\sum t_D}{t}$, i. e. the sum of all down times t_D during time t. The time, that is needed to repair a machine is t_{rep}. t_{rb} is the time, needed to rebuild the machine. t_p is the time needed to produce an workpiece and is subjected to change.
Let T_p^n be the set containing the last n ($|T_p^n| = n$) processing times t_p of the products being produced and $[\max\{t_{p_1}, t_{p_2}, \ldots, t_{p_n}\}, \frac{1}{\sqrt[n]{\alpha}} \cdot \max\{t_{p_1}, t_{p_2}, \ldots, t_{p_n}\}]$ the $(1 - \alpha)$ confidence interval for uniformly distributed data t_{p_i}. With this confidence interval we estimate $t_p^{est} = \min(T_p^n) + \frac{1}{2\sqrt[n]{\alpha}} \cdot (\max(T_p^n) - \min(T_p^n))$.
Let t_{gap}^o be $t_d - t_o$, and let t_d be the remaining time to the deadline, t_{gap}^o specifies

the time, when the execution of the order o must be started. With t_{gap}^o, an order o_i can now be associated with a priority $P_i = \frac{1}{t_{gap}^{o_i}}$. The larger the value of P_i the more important is the order. The orders i are queued according to their priority P_i, which must be recalculated each time step, because with proceeding time, P_i increases. In order to balance the load, the utilization $U_i(t) = \frac{t}{\sum_{t_d^i \leq t} \frac{1}{P_i}}$ of the machine i is used, this is an indicator of the load of the machine at the time t. The value permanently changes. If a new order is added to the machine it increases and if a new piece is produced it decreases. The overall goal in the manufacturing system is, that all machines should have approximately the same utilization. In this case we talk about a *balanced system*. It is the LBAs task to control U of its machine and compare it with U_i's from other machines. If the U_i differ extremely, orders have to be shifted to other machines.

5.2 Determining the Coalition Size

In this section the optimal number n_{opti} of involved machines is determined. Let n be the number of available machines, t_{rb} the rebuilding time for a machine, s the number of pieces the order consists of, and t_p the time a machine needs to produce a workpiece. The time $t_o(n)$, that n machines need to complete the order, is given by equation 2 with the degree of parallelism p[4] $((0 \leq p \leq 1))$:

$$t_o(n) = (n - np + t) \cdot t_{rb} + \frac{s}{n - np + t} \cdot t_p \qquad (2)$$

Equation 2 only holds, if each machine produces the same amount of workpieces $\frac{n}{s}$ and the resetting of machines is not done completely contemporaneously. The last assumption is often true, because machines even if they start producing an order at the same time, do not complete it isochronously, because of breakdowns or variances in the cycle time. The optimal value n_{opti} is obtained by deriving equation 2. This leads us to $n_{opti} = \sqrt{\frac{t_p \cdot s}{t_{rb} \cdot (1 - p^2)}} - \frac{p}{1 - p}$. Finally, there are $\max\{2, \lceil n_{opti} \rceil\}$ machines in a coalition. If the number of pieces is not equally split among the machines, n_{opti} increases. Sometimes it happens, that n_{opti} exceeds the number of available machines. In this case, the negotiation needs not to be carried out, because all machines participate in completing the order.

6 Summary

In this paper we focused on a very important subsystem of manufacturing systems. It is a Machine-Shuttle-Warehouse-Shuttle scenario (MSWS-System). A product is produced by a machine M, transported by one of a set of autonomous shuttles S in warehouse W and eventually transported to another machine by another set of shuttles S. This paper shows the architecture of

---

[4] p equals 1, when an order is done completely parallel and it is 0, when it is done one after another.

a MSWS-System, composed of various agents, all having a well defined functional scope. A load balancing mechanism guarantees, that the load is equally split among all machines even in exceptional cases. Equipping the machine and its operators with profiles, results in a higher degree of self-determination at the working place.

We simulated a MSWS-system with data of from a real tyre factory. First promising results are available, but must not be published at present time.