On applying the PROSA reference architecture in multi-agent manufacturing control applications

Paul Verstraete, Bart Saint Germain, Karuna Hadeli, Paul Valckenaers, and Hendrik Van Brussel

K.U.Leuven, Leuven, Belgium

Abstract. PROSA is a holonic reference architecture for manufacturing control. This paper is inspired by the practical experience of our group in applying this reference architecture in concrete applications. The paper makes two key observations. As a first observation, all entities in the 'world of interest' should not only be represented as an agent in the multi-agent system but also as an entity in the environment. A second observation is that there exist multiple viewpoints on the basic elements in the reference architecture. Two examples of such viewpoints are given. It is explained how PROSA merges both views.

1 Introduction

PROSA, as a reference architecture, provides a template for the architectural design of manufacturing control systems. The architect can select from a standard set of components and uses them in a way appropriate for the desired system architecture. The two following sections give more details about PROSA and reference architectures.

The basic components in PROSA represent three relatively independent aspects of the manufacturing reality. The clear correspondence between the structure of the reality, in this case the manufacturing system, and the architecture of the manufacturing control system makes it easier to adapt the manufacturing control system architecture. Section 5 explains this.

The reference architecture ensures that the important aspects are not left out of the architecture. PROSA does not require the architect to be a specialist in all aspects of manufacturing. Depending on the viewpoint of the architect, some basic components will have mature implementations, while others have temporary ones. Providing all basic components of PROSA in the architecture, allows to extend the architecture to more mature versions later on. Section 6 details this further.

2 A holonic reference architectures for multi-agent manufacturing control systems

This section explains what a holonic architecture for multi-agent manufacturing control systems is. First, it defines architecture in general. Next, it discusses

holonic architectures, a specific type of architecture, used in the context of manufacturing control. Finally, the relation between multi-agent systems and holonic manufacturing control systems is clarified.

2.1 What is an architecture?

In literature there are many definitions for the term architecture (e.g. [17, 19, 4]. In this paper, we use the definition from Wyns [19]. He defines architecture as:

- the art or practice of designing and building structures and especially habitable ones,
- formation or construction as (or as if as) the result of a conscious act,
- architectural product or work,
- a method or style of building.

Architecture may refer to a product, e.g. the architecture of a building, or it may refer to a method or style, e.g. building architecture, the knowledge and styles used to design buildings.

Reference architectures and system architectures are two different types of architecture.

A system architecture is defined as [19]: the architecture of a specific construction or system (e.g. a piece of software, a building or a mechanical device). System architecture corresponds to 'architecture as a product'. A system architecture is the result of a design process, and specifies a solution for a specific problem. It specifies the structure of the solution, the components, and their responsibilities, dependencies, interfaces, data, interactions, and constraints.

A reference architecture [19] corresponds to architecture as a style or method as mentioned in the dictionary. It refers to a coherent design principle used in a specific domain. An example of such an architectures is the Gothic style for building. The architecture may specify one or more of the following aspects: a generic system structure, the kinds of system components, their responsibilities, dependencies, interfaces, data, interactions, constraints, design rules, etc.

The reference architecture is used as the basis for designing the system architecture for a particular system. When designing a system according to an architectural style, the architect can select from a set of standard components and use them in ways appropriate to the desired system architecture [9].

2.2 Holonic architecture

Holonic architectures are composed out of holons. They were introduced in the manufacturing control domain, to create systems that could better deal with current and future challenges then the existing systems.

To discuss holonic architectures further, we define the terms manufacturing control and holon.

Manufacturing control Manufacturing control is the operational level of production planning and control. It is the decision making activity concerned with the short-term and detailed assignment of operations to production resources.

Holon Koestler proposed the word Holon 25 years ago. The word itself is a combination from the Greek holos = whole, with the suffix on which, as in proton or neutron, suggests a particle or part.

Holons are autonomous self-reliant units, in the sense that they have a degree of independence and can handle contingencies without asking higher authorities for instructions. They are stable intermediate forms, that can be used to create a bigger whole. They are self-contained wholes to their subordinated parts, and at the same time dependent parts when seen from the inverse direction.

By using holons, designers aim to create complex systems that can be reused and redesigned much more rapidly.

Holonic architectures Holonic architectures were introduced to provide an answer to some shortcomings in existing architectures of manufacturing control systems. Two types of architecture are frequently used,hierarchical architectures and heterarchical architectures.

Hierachical architectures [4] introduce levels of control and have a pyramidical structure. There are strict master-slave relationships between the components. Control decisions are operated top-down and status is reported bottom-up. Hierarchical control claims to have a high and predicatable performance. Local controllers are helpless however when cut-off from their directing supervisors. This makes robustness very difficult and increases the coupling between the different modules. Also, in hierarchical control, decision making is often based on obsolete information, because of the complexity of the algorithms at the higher levels in the hierachy.

Heterarchical architectures [4] allow for full autonomy between the components in the architecture. The cooperation between entities is arranged via an explicit negotiation procedure. (e.g. contract-net protocol) or via indirect coordination. Full local autonomy is maintained during the cooperative process. This architecture claims to increase robustness. Because components function autonomously, they should not fail when other components malfunction. The disadvantages of this architecture are reduced predictability of the control system and possible incompatibility issues.

Holonic architectures try to combine the high and predictable performance promised by hierarchical systems with the robustness against disturbances and the agility of heterarchical systems by having characteristics of both architectures. A holon can function on it's own (as a whole), which increases robustness (as in heterarchical architectures). At the same time, it can also function as part of a bigger whole, forming a hierarchy with other holons for a certain period of time. This is a feature similar to hierarchical architectures.

2.3 Multi-Agent manufacturing control system

To discuss the relation between holonic architectures and multi-agent systems in the context of manufacturing control, we clarify the terms agent and multi-agent system.

Agent comes from latin, and literally means 'a person who is acting'. In the context of software engineering, it is a computer system, that is situated in some environment, that is capable of flexible autonomous action in order to meet its design objectives [18]. A multi-agent system is a loosely coupled network of problem-solver entities that work together to find answer to problems that are beyond the individual capabilities of knowledge of each entity [7].

A multi-agent system can have different types of architecture, e.g. a heterarchical or a holonic architecture. Agents can serve as components of both architectures, since these architectures contain autonomous components. However, not all architectures are possible. A multi-agent system cannot have a hierarchical architecture for example. A hierarchical architecture has components which are totally dependent towards their supervisors [4], which is not compatible with the notion of an agent.

This paper discusses the application of a holonic reference architecture, PROSA, to a particular type of systems, multi-agent systems, in a particular domain, manufacturing control.

3 The PROSA reference architecture

PROSA is a holonic reference architecture for manufacturing control. PROSA stands for Product, Resource, Order and Staff holon. The product, resource and order holons are mandatory components of the architecture. The staff holon is an optional component.

The reference architecture is designed to [19]:

- separate the necessary components, which are generic, from the optional components, which can be manufacturing system specific
- separate the structural aspects of the architecture from the algorithmic aspects for resource allocation and process planning
- separate the resource allocation aspects from process specific aspects
- foresee migration and evolution possibilities to enable the incorporation of legacy systems, or the introduction of new technology.

More information on the reference architecture and its components is given in the following sections.

4 Two key observations in applying the PROSA reference architecture

Our group at PMA has been applying the PROSA reference architecture in concrete applications for many years. This paper discusses two key observations, based on this experience:

– *representing entities in the 'world of interest':* The components of the system archictecture represent something that exists in reality (i.e. that part of reality that is relevant for the application). This allows the control architecture to be adapted to changes in that reality more easily (new workstations, additional orders, fluctuating transportation delays,...)[12],
– *multiple viewpoints on the basic elements in the reference architecture:* people in a factory have to deal with many problems at once. The personal experience and background determine which of these problems is considered the most important. Therefore, a manufacturing control architecture (and a manufacturing control reference architecture), is required to support heterogenous requirements.

Both observations help to reveal why the product, resource and order holon are the basic components of the PROSA reference architecture. They represent three relatively independent aspects of the manufacturing reality. These aspects are important in every application of the reference architecture in manufacturing, even if they are perceived differently depending on the viewpoint.

5 Representing entities in the 'world of interest'

This section presents a holonic architecture for multi-agent manufacturing control systems. First, the architecture itself is discussed. Next, it is shown how the three basic holon in the PROSA reference architecture (order, product and resource) fit into this architecture. The final section discusses which parts of the architecture are vulnerable to changes, why they are vulnerable and how this affects the total architecture

5.1 A concrete holonic architecture for manufacturing systems

In this section, we give an overview of the different components of the manufacturing system architecture, their responsibilities and interrelationships (as shown in Figure 1). A holonic manufacturing system(HMS) comprises both the manufacturing control system and the manufacturing system itself [13]. At the highest level, the archictecture therefore consists out of two layers:

– the control system,
– the world of interest.

Holons are components that crosscut the control system and the world of interest. Holons have an optional (emulated) physical part (e.g. raw material, machine) which belongs to the world of interest. They have an information processing part and decision taking part which belongs to control system.

In the following paragraphs, we discuss the architecture and responsibilities of each of these components.

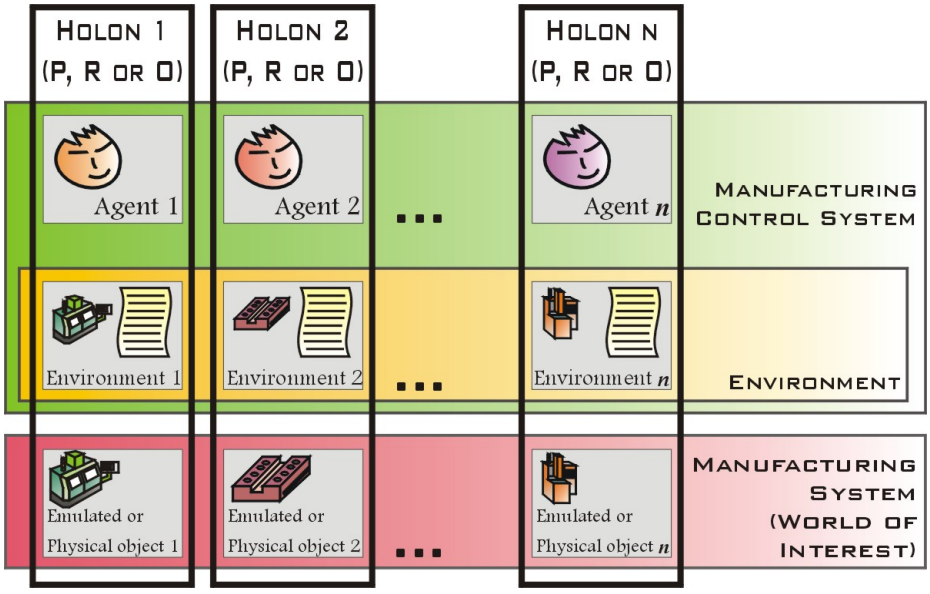

Fig. 1. A holonic architecture

Control System The control system is a multi-agent system, which treats both the environment and the agents as first order abstractions. In this section, we define the environment. Next, we discuss the responsibilities of the environment and the agents.

Environment The environment is a software system which represents the world of interest in the manufacturig control system. It provides:

- *an emulation model of the manufacturing system:* The environment contains emulation models of the entities (e.g. machine, tool,) in the world of interest, i.e. models that mimic the behaviour and structural aspects of those entities,
- *specific knowledge of the resources (e.g. machine, tool, ...):* The environment is not limited to providing sensory information to the control system. It can also interpolate this information (e.g. interpolating the position of a transporter between two sensor positions) or provide support for what-if scenarios [11].

In this architecture, the environment has the following responsibilities:

- it provides a structuring to the MAS and enforces physical rules in the environment,
- a consistent interface to the manufacturing system (see Figure 1). The environment provides one consistent shared data model, representing the manufacturing system,

– it serves as a communication infrastructure for the agents. The environment provides a blackboard infrastructure that enables indirect interaction through stigmergy [10, 11].

Agents An agent has the responsibility to fullfill a goal (which can be a going concern [8] or a goal that just needs to be accomplished once). The agent is provided the decision rules to select the best option for fullfilling its goal.

In this architecture, each type of agent has a different responsibility. An agent can be an order agent, a product agent or a resource agent.

– *order agents* normally seek to accomplish their task. These agents search for solutions and select amongst candidate solutions the most attractive one to become their intention, which eventually gets executed,
– *product agents* provide order agents with knowledge about available routings and processing sequences that are able to create the proper product,
– *resource agents* drive their resource (actuation) and keep their internal state model synchronized with the resource through appropriate input (sensor readings).

The World of Interest The world of interest (WOI) is that part of reality which falls within a certain scope relevant for the application.

According to Webster scope is defined as: An area in which something acts or operates or has power or control: 'the range of a supersonic jet'; 'the ambit of municipal legislation'; 'within the compass of this article'; within the scope of an investigation'; 'outside the reach of the law'; 'in the political orbit of a world power'.

Scope in this context has two dimensions:

– *range:* That part of the world in which the manufacturing operates,
– *level of detail:* The smallest part which is relevant for the application.

Holon First, we discuss the architecture of the holon. Next, the responsibilities of the various holon types (resource, order and product) are discussed.

Architecture of a holon The architecture of a holon in general contains the following components [1]:

– *physical processing component:* the actual hardware performing manufacturing operations or an emulation model of that hardware,
– *physical control component:* a controller making sure the hardware stays in a safe state and is capable of performing basic commands,
– *decision making component:* this model contains the decision rules the holon uses,
– *inter-holon interface:* an interface for interaction with other holons,
– *human interface.*

In this architecture:

- the physical processing component of the holon is part of the world of interest,
- the physical control component, as far as the control stays decision free, together with an emulation model of the actual hardware are part of the environment,
- the decision making component is part of the agent.

That part of the holon that is also part of the environment is called the environment entity. The environment entity provides the following functionalities:

- *information about the physical part:* In this model all properties of the physical part are included that are important or have been important in the past in the operation of the resource, even if they were important only once. They are properties of the resource as a whole, not properties of it's internal parts.
- *information about the present state:* The entity in the environment synchronizes it's state with the real world. It keeps track of the events that are relevant to the resource as a whole. Which information can be delivered to the entity depends on the information that can be delivered by the real world. The environment entity can also interpolate information between events.
- *information about the near future:* Agents can deposit state information into the environment entity about the expected states in the near future. Environment entities are not capable of generating this information themselves, since they should not be aware of the objectives of the agents. They can however store this information.
- *what-if functionality to the agents:* Environment entities can give information about the properties of their state in the future, if this information is delivered by agents. This allows for other agents to estimate the state of the environment in the near future. A more detailed explanation be found in [11, 6] .

To summarize: a holon = agent + environment entity + physical (or emulated) hardware.

Responsibilities of the various holon types A holon has both the responsibilities of its environment entity component and of its agent component.

In PROSA, the three type of holons have the following responsibilities:

- *resource holon:* A resource holon offers production capacity and functionality to the surrounding holons. It contains a physical part, namely a production resource of the manufacturing system, and an information processing part that controls the resource.
- *order holon:* An order holon represents a task in the manufacturing system. It is responsible for performing the assigned work correctly and on time. It manages the physical product being produced, the product state model, and all logistical information processing related to the job.

– *product holon:* A product holon holds the process and product knowledge to assure the correct making of the product with sufficient quality. A product holon contains consistent and up-to-date information on the product life cycle, user requirements, design, process plans, bill of materials, quality assurance procedures, etc. As such it contains the product model of the product type, not the product state model of one physical product instance being produced.

5.2 Discussion

A holon in the context of manufacturing systems mimics the 'structure of the problem domain' (the production facility) into the architecture of the manufacturing control system. This is done by enriching every physical entity (in the WOI) with an environment entity component and a decision taking component.

If the structure of the manufacturing system evolves, the clear correspondence between the structure of the manufacturing system and the architecture of the manufacturing control system makes it easier to adapt the manufacturing control system architecture.

A holon increases separation of concerns between the domain entities. By gathering all components related to one domain entity into one component, this component can be reused everywhere where the domain entity is needed (e.g. a machine manufacturer delivers resource holons with each machines which can be integrated into the manufacturing control system of the client. In the case of PROSA:

– Resource holons only need knowledge about their own resource. Logistical concerns and product related concerns (e.g. sequence of operations) are shielded from the resource holon.
– Rrder holons only need knowledge about the raw material they govern and the task they have to fulfill. They are shielded from technological concerns (product) or detailed knowlege of the resources.
– Product holons only need to be aware off process and product knowledge. They are shielded from logistical concerns(order) or the concrete resource allocation policy (resource) used in the plant.

6 Multiple viewpoints on the basic components in the reference architecture

As stated before, PROSA has three basic elements: the order holon, the product holon and the resource holon. Experience shows that, depending on the background of the 'designer', these three basic elements are perceived differently. To illustrate this, we take an industrial example (based on reality but simplified), which is described in a first section. For this example, we describe two viewpoints, namely the viewpoint of a product engineer and of a production manager. After describing their viewpoints, we give an example high-level design of the manufacturing control system. Finally, we discuss how this design

can fullfill the requirements of both the product engineer and the production manager.

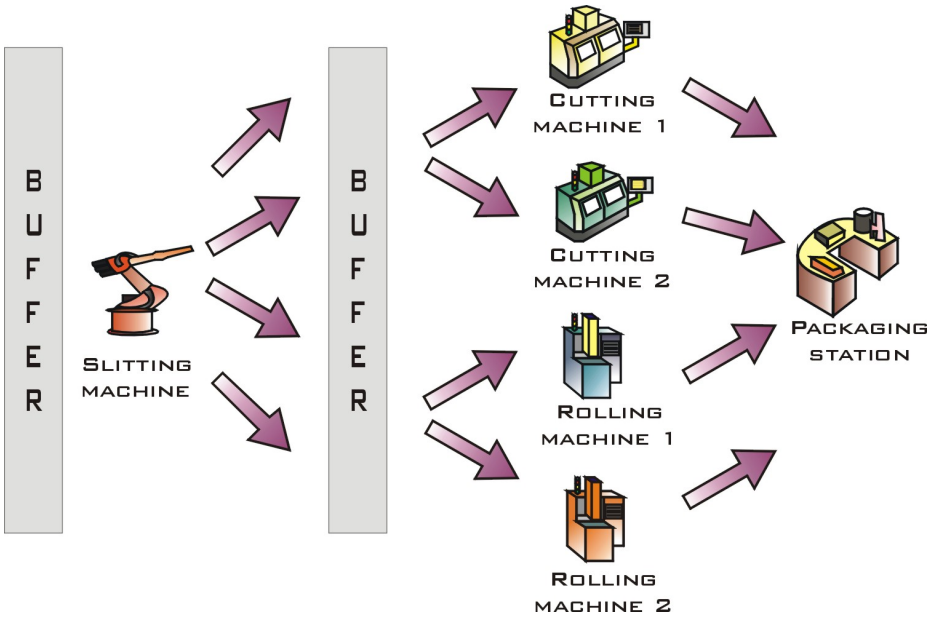

Fig. 2. An example industrial case

6.1 An example industrial case

The industry discussed in this example produces optical film for medical applications or for use in safety checks of sewage systems. It has the following characteristics:

- *large product volumes together with an increased product variety:* The clients are mainly other businesses, requesting large product volumes. However, demand for the main product has dropped over the years. Niche products have been released to the markets, thus increasing the product variety. More and more small quantities of specialized products are requested together with larger orders.
- *expensive raw materials:* Silver is a very important ingredient of the film. It directly influences the film quality and is very expensive. Therefore it is crucial that the appropriate amount of silver is present in the film. Too much silver is too expensive, not enough delivers bad quality film.
  From time to time, the film is damaged at some spots (traces on the film). These parts should be recycled as much as possible. This depends on the

type of film and the quality required. Also new parts, in low quantities, need to be combined with other orders, to replace the scrap.

- *unconventional process plans:* Standard production planning software (e.g. MRP) expects the product to be assembled out of smaller parts. In this case, the product is disassembled out of raw materials. A big roll of fime, called the 'master roll' is slit into smaller pieces by a so called 'slitting machine'. After slitting, the films need to be cut into leaves in 'cutting machines' or are rolled on 'rolling machines'. After that, the film 'leaves' or the film 'rolls' are packaged and put in pallets, after which they are transported to the clients.
- *delicate production processes:* The film has week spots, imperfections caused by production. These imperfections should be evaluated. The decision whether or not a certain type of film can still be produced with such a spot depends on the specifications of the particular type of the film.

  Some machines are more accurate than others. Older machines can theoretically perform the same operation as the newer ones, but for high quality film, the newer machine is preferable. In cases where the newer machine is not available or not functioning well, the older machine can still be used, but special care should be taken. High quality products produced by older machines should undergo additional quality checks.

This example discusses one production facility, which is used by both company divisions. The topology of the production facility is depicted in Figure 6.It has one slitting machine which serves two production lines. The one production line mainly consists out of the older machines, while the other production line has the newer ones. For some types of film, only part of the new line can be used. The last operation steps are performed on old machines. If a machine in one of the two lines fails, the corresponding machine in the other line can be used.

6.2 Two viewpoints on the industrial case: a product engineer and a production manager

The production manager is responsible for optimizing the day to day production of a production facility.

The production manager is bound by the master production schedule which states which quantities of which type of products have to be produced on a monthly basis. If the client changes his mind within this period, he is responsible for resolving this conflict.

He can resolve a conflict by adjusting the production plan, by introducing new orders or by cancelling orders.

If production passes duedate, this is his responsibility. Sometimes he has to introduce rush orders (which have priority over all other work) to satisfy an important inpatient client.

The product engineer is responsible for producing high quality optical film in the same production facility. He has a large technical knowledge about the product and the machines available in the plant. He has no direct contact with clients. Quality complaints or complaints on the design of the product are his concern.

6.3 An example high-level design

The design starts with gathering the requirements from all parties involved, in this case the production manager and the product engineer. In a next step, these requirements are used to determine the scope of the application. This scope determines the relevant components for the environment that is modeled in the following step. Finally, some suggestions for possible decision algorithms (used by the agents) are given. The selection of the appropriate decision algorithm is also driven by the requirements.

Requirements The production manager needs to satisfy clients by delivering their orders in time. Therefore he needs the following information:

- *up-to-date status reports of orders:* Clients request up-to-date information on the progress of their orders. To be capable of delivering this information to the clients, the production manager needs to be capable to trace the production in the plant.
- *short-term forecasts of order progress:* Clients want dependable delivery times. A client needs to organize its own activities (production or private use). Unexpected delays make this more difficult. Therefore it is important that there are as few delays as possible. If a delay cannot be avoided, it should be communicated to the client as soon as possible, and the estimated delay should be reliable. By doing this, the client gets maximal chances to adapt to the unforeseen change [5]. To be capable of delivering this information to clients, producing short-term forecasts on the delivery are part of the core business of the production manager.
- *flexibility to deal with unavoidable disturbances (e.g. late deliveries, sickness, breakdown of a machine):* The production manager needs to be capable to perform adaptations quickly. The consequence of this adaptations should be communicated to all parties involved as fast as possible.

The product engineer needs to satisfy future or current clients by producing high quality optical film. Therefore he needs the following information:

- *up-to-date technological knowledge:* A product engineer has very specific knowledge on the capabilities of the different workstations and the processes in the plant. This knowledge is constantly evolving: process plans are refined, workstations are improved, new products or processes are introduced and new machines are bought.

- *quality control:* A product engineer wants information on the currently obtained quality, and the evolution on the quality in the near future. The quality requirements are subject to evolution (new possibilities because of new processes, higher competition, ...).
- *maintenance:* While not being directly involved in maintenance, this activity has direct impact on the product quality. Therefore, current maintenance activities and in the short term future should be visible for the product engineer. He should also be capable of influencing those activities. Not directly responsible for maintenance, still might have an impact

Scope As mentioned before, scope in this context has two dimensions:

- *range:* Part of the world in which the manufacturing operates.
- *level of detail:* The smallest component which is relevant for the application.

The range in this case contains the production facility, the clients and the raw materials. The level of detail is determined by influences the designer want to excert on the manufacturing process and the information he wants to obtain. Every concept needed to express these should be explicitly included in the architecture.

- *The production manager wants to influence:* the release of orders and the logistics of the order.
- *The product engineer wants to influence:* the product design, on which machine an operation is performed, the operation settings used for performing an operation and finally the quality inspection of the product.
- *The production manager needs information on:* load forecast, client status, breakdowns, alternatives for assigning operations to machines.
- *The product engineer needs information on:* outcome of operations, operations to be performed, current state of the machine and it's capability.

Note that this is only an illustrative analysis for determining the scope. In reality a more detailed analysis is made.

The environment The environment is composed out of environment entities. Which entities are relevant is determined by the scope of the application (see previous section). The detailed modeling of an environment entity is out of the scope of this paper [16]. Instead this section discusses the information provided by each of the three different entity types: resources, orders and products.

- resource type entities provide information on:
  - *the model of the real part:* There are three types of resources: slitting resources, cutting resources and turning resources. Each type of resource delivers information about it's capabilities. E.g. for a slitting machine, it delivers the range of accuracy in which it can slit (perfectly maintained or completely unmaintained), the speed range it supports for slitting, ...

  - *the present state:* This information comprises the properties of the states, their current values and the range of values between which these properties can vary using information about the current state. These ranges are subset of the ranges present in the model of the real part. E.g. theoretically a new slitting machine can deliver a speed between 100m/s and zero,but because of calibration problems the speed range has diminished to 84m/s and zero. The current speed is 82m/s.
  - *the near future state:* This information comprises the properties of the states and range of values between which these properties may vary in the near future. An entity can store this information optionally, after receiving it from the agents. E.g. Because no maintenance is expected to be performed in the near future, in two days, the slitting machine will have a speed range between 83.7m/s and zero.
- product type entities provide information on:
  - *the model of the real part:* In this model all relevant properties of the product type are included. An example of product type is a film in the 80 $\times$ 104 mm format, leaves. This model contains the technically possible ranges for the properties of this product type. For example, a film in the 80x104 mm can contain from 50 to 70 mg silver. Less or more silver makes the film useless. That 70 mg silver is far too expensive, and does not provide a quality gain, is a choice related to the objective of the agent (good quality at an economic price) and therefore not relevant in this model.
  - *the present state:* There is a difference between technically possible and technically realistic values. For the 80x 104 mm film, there is no raw material to be found that contains more than 60 mg silver. Therefore, independent of what the policy is of the production facility is at this moment, 80 $\times$ 104 mm with more than 60 mg of silver is technically possible, but not realizable because of the unavailability of the raw material.
  - *the near future state:* What is technically realisable and even what is technically possible changes over time. If information about evolution in this information is available for the near future, agents can store this information in the entity. For example, raw material with silver up to 63 mg silver is expected to be delivered in the near future, because of market demand. Therefore, up till 63 mg silver film of the 80 $\times$ 104 mm film type will be technically realizable in the near future.
- order type entities provide information on:
  - *the model of the real part:* In this model all relevant properties of a ordered product are included. An example of an order type is a film in the 80 $\times$ 104 mm format, leaves, due in two weeks, ordered by an important client. This model contains the desired property ranges of the client (which should be within the technically possible ranges) For example, the ordered film should have a silver level between 54-56 mg silver. 55 is the actually desired value, but a small deviation is tolerated by the client.

- *the present state:* This information contains the current state of the product ordered by the client. It could still be raw material, part of a bigger roll, at this point in time, lying somewhere in the storage.
- *the near future:* Agents can store information about the near future in this entity. For example, the order is expected to be slitted in about 5 hours.

The agents A detailed design of the agents (specifying the roles of the agents and their design) is out of the scope of this paper. Instead, example decision rules will be given for each agent type.

- *resource agent:* The resource agent can decide which orders to produce at what time. He is given a schedule that he tries to execute [14, 15]. If this schedule turns out to be unfeasible, he uses dispatching rules to decide which operation to execute first. Example dispatching rules are First Come First Serve(FCFS), Earliest Due Date (EDD), shortest production time (SPD) [2].
- *order agent:* An order agent is responsible for getting an order produced. He is responsible for searching alternative solutions and can consult the product agent for advice on the technical feasibility and the desirability of the alternative. He decides which alternative to use for producing an order. The order agent is initialised with a solution to get his order produced. If this solution is feasible, he executes this. In the meanwhile he keeps searching for possible alternatives. If the solution becomes unfeasible (because of breakdown, late delivery) he uses a perfomance indicator to compare the alternatives he found, and to choose a new solution to execute. Example performance indicators are leadtime or work in progress.
- *product agent:* A product agent holds the knowledge to produce a product type. More specifically, he needs to decide which operations are technically suited for which machines and which sequences of operations can be performed. To determine the possible sequence of operations a non-linear process plan is used [3]. The product agent is also provided with rules to determine which capabilities are suited for which operations (e.g. slitting speed should be higher than 50m/s).

6.4 Discussion

In the previous section, we presented an example high-level design for an industrial example. This section examines each requirement and discusses how it influences the presented design.

The production manager requires:

- *up-to-date status information on the progress of the orders:* Status information is encapsulated in the environment entity. The complexity of this information does not affect the complexity of the calculations the agents make in using their decision rules. The agents extract from the environment only the information they need for their calculations. It does not affect the

reusability of the agents either. The environment is responsible for synchronization with the reality. This relieves the agents from this responsibility and avoids that they need the knowledge to update these models. Because adding status information only affects the environment entity and not the rest of the system, it is easier to keep this status information up-to-date: all relevant aspects can be reported easily and no simplifications need to be introduced for complexity reasons.
- *short-term forecasts of order progress:* Order agents have information on the solution they are intending to execute (this solution can be provided to them or they can search for it themselves). They can store this information in the environment entity. This creates short-term forecasts of the order progress. [11]
- *flexibility to deal with unavoidable disturbances:* If a breakdown occurs, all order agents will search for an alternative themselves. This causes the system to self-organize and adapt to the change [11]. Note that this adaptation can take unavailability of workers into account because they are modeled in the environment, so that up-to-date status information of the workers is available. The production manager can (but is not required to) influence the change process by providing solutions to order agents. He can also modify the decision rules used by the order agents.

The product engineer requires:

- *up-to-date technological knowledge:* Technological knowledge is encapsulated in the product holon. Evolution in technological knowlegde only requires this holon to be adapted. This reduces the effort to upgrade the manufacturing system and increases the chance that up-to-date knowledge will be used by the manufacturing control system in its operation.
- *control on quality:* Because up-to-date status information of the orders is available and this information can be very detailed, quality can be easily inspected by the production manager.
- *influence on maintenance:* maintenance schedules can be easily stored in the environment entities. In this manner, this information is visible to all orders so that they can take it into account. The product engineer can modify or add to this information.

7 Conclusion

The PROSA reference architecture provides a generic solution for designing manufacturing control system architectures. It contains knowledge on the manufacturing domain. This knowledge ensures that no important aspects of the manufacturing domain are omitted in the architectural design

By presenting a high-level system architecture, the paper reveals why the three basic components of the architecture are chosen to represent three important aspects of manufacturing reality. The high-level system architecture consists

out of holons. A holon contains an element in the world of interest, an environment entity and an agent. Every element in the world of interest, in the scope of the application, corresponds to a holon. The holonic architecture thus mimics the structure of the physical manufacturing system in the structure of the manufacturing control system. If the structure of the manufacturing system evolves, this clear correspondence makes it easier to adapt the manufacturing control system architecture.

The paper also illustrates how the three basic components of the architecture can be perceived differently, depending on the background of the designers. They are all three relevant in every design. The production manager, since he is mainly dealing with logistics and resource allocation, considers the order and resource holons as the most important components, and can provide detailed input for their implementation. The product engineer has more knowledge on the product holon. To also fulfill his requirements, it is important that more advanced version of product and process planning integrate easily into the architecture (e.g. introducing non-linear process planning).

References

1. Bussmann, S. An agent-oriented architecture for holonic manufacturing control. In *Proceedings of First International Workshop on IMS* (1998).
2. Daniel Sipper, R. B. *Production: Planning, Control and Integration.* McGraw-Hill, 1997.
3. Detand, J. *A Computer Aided Process Planning System Generating Non-Linear Process Plans.* PhD thesis, Katholieke Universiteit Leuven, 1998.
4. Dilts, D. M., Boyd, N. P., and Whorms, H. H. The evolution of control architectures for automated manufacturing systems. *Journal of Manufacturing Systems 10* (1991), 79–93.
5. Germain, B. S., Valckenaers, P., Verstraete, P., and Van Brussel, H. Resource coordination in supply networks. In *SMC (2)* (2004), pp. 1972–1978.
6. Holvoet, T., and Valckenaers, P. Exploiting the environment for coordinating agent intentions. In *AAMAS conference 2006, 8-12 May, Hakodate, Japan* (2006).
7. O'Hare, G., and Jennings, N. *Foundation of Distributed Artificial Intelligence.* Wiley Inter-Science, 1994.
8. Parunak, H. From chaos to commerce: Practical issues and research opportunities in the nonlinear dynamics of decentralized manufacturing systems. In *Proceedings of Second International Workshop on Intelligent Manufacturing Systems* (1999), pp. k15–k25.
9. Perry, D. E., and Wolf, A. L. Foundations for the study of software architecture. *ACM SIGSOFT Software Engineering Notes 17* (1992), 40–52.
10. Theraulaz, G., and Bonabeau, E. A brief history of stigmergy. *Artificial Life 5*, 2 (1999), 97–116.
11. Valckenaers, P., and Van Brussel, H. Holonic manufacturing execution systems. *Cirp Annals-Manufacturing Technology 54*, 1 (2005), 427–432.
12. Valckenaers, P., Van Brussel, H., Bochmann, O., Saint Germain, B., and Zamfirescu, C. On the design of emergent systems: an investigation of integration and interoperability issues. *Engineering Applications Of Artificial Intelligence 16*, 4 (2003), 377–393.

13. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., and Peeters, P. Reference architecture for holonic manufacturing systems: Prosa. *Computers In Industry 37*, 3 (1998), 255–274.
14. Verstraete, P., and Valckenaers, P. Towards cooperating planning and manufacturing execution systems. In *12th IFAC Symposium on Information Control Problems in Manufacturing,17-19 May, Saint Etienne, France* (2006).
15. Verstraete, P., Valckenaers, P., Van Brussel, H., and Hadeli, K. Integration of planning systems and an agent-oriented mes. *International Journal for Information Technology and Management 8*, 1-3 (2006), 159–174.
16. Verstraete, P., Valckenaers, P., Van Brussel, H., Saint Germain, B., and Hadeli. Engineering environments for flexible and easy to maintain multi-agent manufacturing control systems. In *The Modern Information Technology in the Innovation Processes of Industrial Enterprises* (2006). to be published.
17. Weyns, D., Schelfthout, K., and Holvoet, T. Architectural design of a distributed application with autonomic quality requirements. In *ICSE Workshop on Design and Evolution of Autonomic Application Software, DEAS, St. Louis, USA, May 21* (2005).
18. Wooldridge, M., and Jennings, N. R. Intelligent agents: Theory and practice. *Knowledge Engineering Review 10*, 2 (1995), 115–152.
19. Wyns, J. *Reference Architecture For Holonic Manufacturing Systems: the Key to Support Evolution and Reconfiguration.* PhD thesis, Katholieke Universiteit Leuven, 1999.