First International Conference
Modelling and Development of Intelligent Systems
Sibiu - Romania, 22-25 October, 2009

Service Oriented Architecture Considerations for Distributed Intelligent Control of Modern Manufacturing Systems

Gabriela Varvara, Carlos Pascal, Doru Pănescu

Abstract

The paper presents some architectural issues on the implementation of software systems designed to support the control of modern manufacturing processes. In the introduction some challenging aspects of modern manufacturing and market problems are pointed out. Then the concepts of an architectural manufacturing paradigm, based on dynamic recursive organization of holons for reconfigurable systems, are synthetically presented. In order to implement control systems that support the holonic recursive decomposition, the authors add to classical multi-agent distributed implementation the Service Oriented Architecture (SOA) characteristics. This development is justified by the necessity to sustain the dynamic cooperation among holons identified during the modeling decomposition phase, as exemplified in the paper, and, also, must be adapted to the characteristics of manufacturing systems as illustrated by the proposed architecture.

1 Introduction

1.1 The Modern Manufacturing Systems – problems, challenges, scenarios

The 21st century market problems required significant changes within the manufacturing systems and the development of associated emerging technologies. Some critical business aspects that need to be addressed are worthwhile to be mentioned [3]: the regular reactivity to "rush orders" and new arrived specification from the customer, an active balance between volume and variety of the production within a single shop floor, the demand for short delivery times for customer-specific products and the need to tightly integrate all the supply chains to hold minimal reserve stocks. All these problems result from everyday plant scenarios and imply a specific response from the manufacturing system as detailed in the next paragraph. The "rush order" reactivity scenario has some characteristics: a rush order can be introduced at any time, has electronic financial credit to buy manufacturing services, is able to negotiate with the existing orders to gain access to services as soon as possible and behaves in order to maximize its profit limits. A mass customization scenario involve flexible choice to meet together different orders that reflect changing of customer requirements with minimizing the stocks (warehouse and out). Another usual scenario refers to the existence of tightly integrated supply chains; the manufacturing system behaves in a proactive manner to find the optimal solution to outsourcing some production by maximal utilization of available resources through cooperation for real-time reaction to changing demands. The above mentioned scenario is tightly coupled with another one

devoted to a high degree of agility and flexible reconfiguration in order to face a great variety of customer demands and/or resources breakdown. This means the capacity to coordinate different resources with their various facilities, multiple and changing ways to make a product, both low volume high variety and high volume low variety manufacturing in the same environment and regular operations of adding, removing or reconfiguration of production machine resources.

In order to deal with these new complexity challenges, the well known centralized solutions to control factories have major drawbacks namely slow reactivity, generation of operational bottlenecks and finally reaching of critical failure points. Consequently, the domain research efforts were focused to develop flexible, fault-tolerant control mechanisms. The result was a new technology that combines both real-time control strategies and deliberative distributed information processing. It brings a new vision on manufacturing process founded on a complete decentralization of the control. As a result of its popularity among the domain researchers it becomes a paradigm of agility named "Holonic Manufacturing Systems (HMS)".

1.2 Holonic Manufacturing Systems – architectural paradigm for modelling of modern production processes

The HMS concepts address all the critical challenges of modern manufacturing and its declared purpose is to develop architectures and technologies for open, distributed, intelligent, autonomous and cooperative systems. As stated by the HMS consortium, there are a few concepts that govern the holonic vision, namely [9]:

- The holon, as an autonomous and cooperative building block of manufacturing systems for transforming, transporting, storing and/or validating information and physical objects. The holon has always an information processing part and often a physical processing part. It is a recursive structure that can be modeled as a tree structure; the nodes from intermediate positions are holons that contains other holons until no more decomposition is possible for the holons from the leaf nodes.
- The autonomy defined as a capability of a holon to create and execute its own plans and to maintain its functionality.
- The cooperation viewed as a process of the execution by a set of holons of mutually acceptable plans.
- Self-organization as the ability to make dynamic configurations of holons addressing a specific production goal.
- Holarchy – the form of holon organization founded on basic rules of cooperation that act in the direction of achieving a goal or objective. This entity defines the limits of holon autonomy.

As a consequence, the holon structure provides a decentralized "bottom-up" approach to develop complex manufacturing systems that ensure a high responsiveness and an agile behavior to changing production environment. Moreover, the holonic concept covers to a certain degree an open model, intelligent and prepared to participate in negotiation processes.

In order to apply the above abstract concepts, it was adopted one of the most used generic architecture PROSA [2] that proposes three different types of holons: order (OH), product (PH) and resource (RH).

The order holons manage the production requirements originating from direct customer demands or other external bodies such as a company/department situated upstream in the supply chain, or even anticipated production needs. They assign priorities; give the options to achieve the current demands by loading the local manufacturing resource holons or outsourcing partially or totally the job.

Resource holons provide all the generic active entities from a manufacturing system such as machines, transportation/storage devices, visual inspection stations or even independent companies.

On the other hand product holons provide knowledge on how to achieve manufacturing objectives, production best recipe/processing guides from multiple options and can disseminate information among other holons; they can offer sometimes and to some degree expert advice.

For complex expertise an expert holon type (EH) with an appropriate architecture can be defined too. As a particular type, a configuration holon (CH) can be created [9] for systems capable of changing their configuration to adapt for external order changes or to internal disturbances. It selects, based on the process plan received from the PH, from more candidate resource configurations capable to accomplish the order, the most efficient one. It is an iterative process based on reconfiguration costs. As soon as PH will be informed by CH about the current configuration, it will send appropriate process information to the RHs. For manufacturing contexts defined by rapid order changes and/or frequent resources breakdown, the RH state will change dramatically. The CHs will be informed about these new states and reconfiguration plans will be started. The optimal resource configuration founded will be sent to OH that, based on the updated product information received from PH, will transmit a new order to all the RHs from the current configuration.

2 Decentralized control and cooperation in HMS

HMS vision proposes [10] for the entire manufacturing system the holarchy model obtained by dynamic aggregation of individual cooperation domain. For example, the cooperation domain of an order holon requests tasks to product and possible resource holons. The entities that are interested to cooperate for achieving the assumed task interact inside a new established cooperation domain. The process continues until no manufacturing tasks can be identified. In order to illustrate this aggregation concept the next assembling scenario it is considered: the goal refers to production of two stacks of pallets, the first containing 3 pallets and the second with four pallets. The same pallet type can appear in different stacks. The following diagram depicts as a tree the dynamic recursive cooperation domains for the holarchy of the proposed manufacturing process (see Fig. 1). The higher level of cooperation for the proposed holarchy [8] is made as a consequence of receiving a manufacturing order from the customer and it is the result of the associated order holon named OH1. The holon entities that accepted to cooperate inside this domain are the product holons for each stack and the additional resource holons RH_1, RH_2 involved in stack transportation from/to loading/storage positions. At the second level of cooperation, each stack product holon will try to identify and aggregate the resources necessary for building the stacks according to their structure (stack I has three pallets of type A, B, C put in a predefined order, etc.). Product holons for pallets A, B and C will agree to cooperate for accomplishing this task. Then the raw part fill-in process for the pallets, according to assembling rules and known recipes, will impose the creation of the third cooperation level domain. It can be noticed that these are the most interior cooperation levels and are based on the participation of only resource holons as the active manufacturing entities of the system.

The configuration holons CH_1, CH_2 and CH_3 will be generated for each level of the holarchy: CH_1 is necessary to establish the resource configuration for assembling the different stacks of the product, CH_2 will be involved in stack formation and CH_3 chooses the configuration for the pallet filling process. The achievement of the global goal will generate configuration holons at different levels of the holarchy. They are dismissed when a current mission finishes and vanish when a reconfiguration process starts.

The cooperation among all the holons inside the holarchy imposes a limitation of the holon autonomy; only the mutually acceptable plans will be started in order to accomplish the global goal. All the cooperation domains along the established holarcy are virtual and dynamic. The connection arrows are dynamically created. They will be implemented using service invocation

mechanism considered to be an appropriate approach. Any task alteration or introduction of new resources will impose real-time reorganization of production resources in different cooperation domains in order to fulfill the customer request.

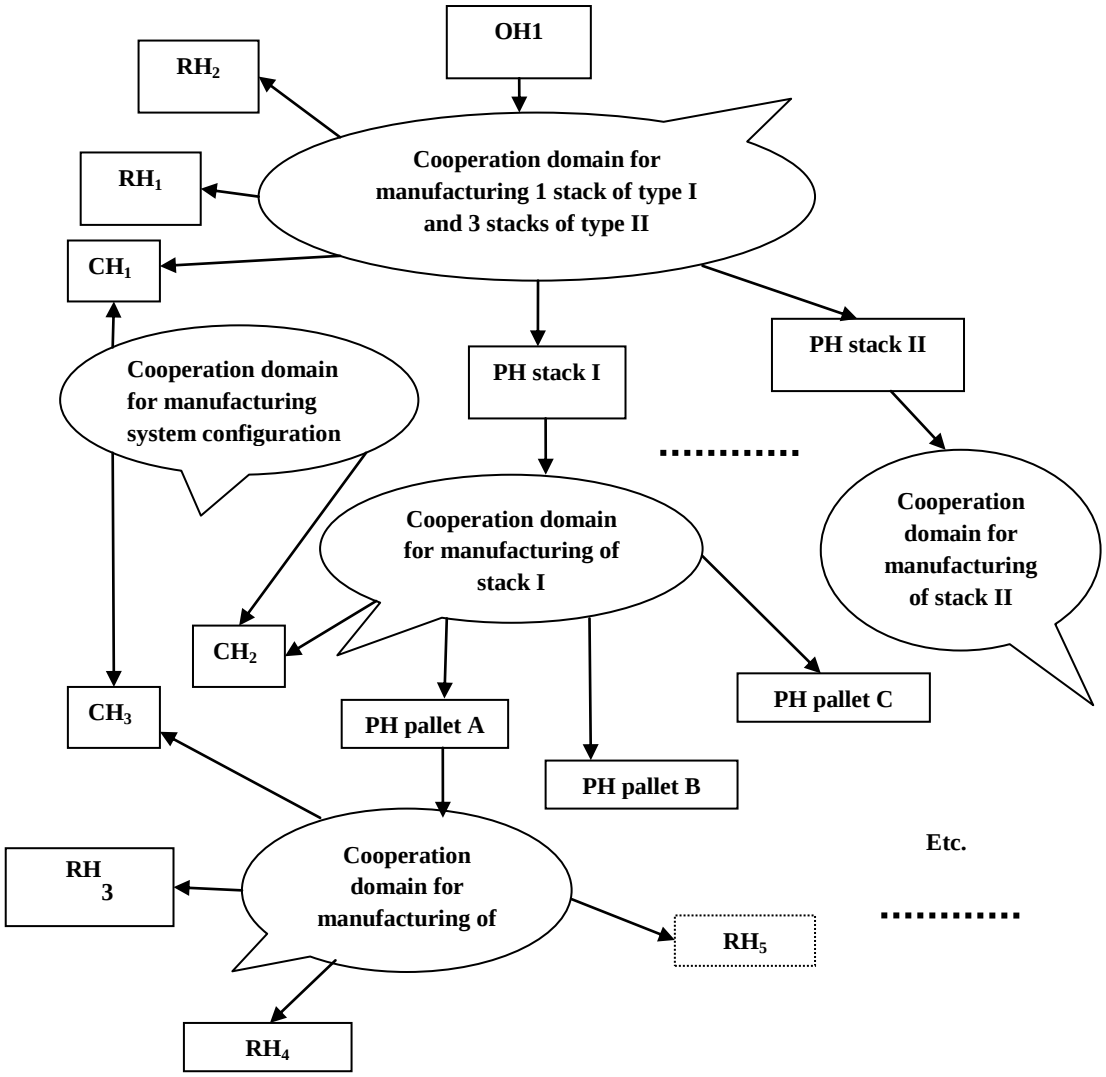

Fig. 1 The Cooperation Domains inside the Holarchy

The decentralization proposed by HMS is based on dynamic creation, breakdown and reconfiguration of virtual cooperation domains. This must be sustained by simultaneous decoupling of control and information processing. The key aspect of such an approach is to separate the planning, controlling, coordination, execution and communication mechanisms for each cooperation domain of the holarchy.

As illustrated in Fig. 2, dynamic cooperation domains will be created along the holarchy for each specific activity. They result as a collaboration agreement among heterarchically organized holons PH, and RH_1 to RH_n. Applying this model to the example from Fig. 1, "PH pallet A" knows the strategy to fill in with parts the pallet of type A. Its goal is to initiate a manufacturing process according to the above mentioned strategy. Consequently, it will initiate a collaboration domain with all the available resource holons able to assume palletizing goals. For instance, at a given moment the pallet assembling plan will impose the attainment of the sub-goal of

transferring of a raw/processed part. The PH will consult all the RHs available for this task. On the other hand the appropriate resource holons will start a planning phase devoted to the selection of an own appropriate plan to realize the transfer operation. It will often imply collaboration with other resource holons that agree to help for reaching the system goal by activating the visual inspection system, other manufacturing cells, resource/part assignment for the assumed operation and the evaluation of the operational costs. All the resource offers will be analyzed by the PH during its own planning phase [8] and finally the most efficient resource holons will be charged to execute the transfer and an execution phase will start.

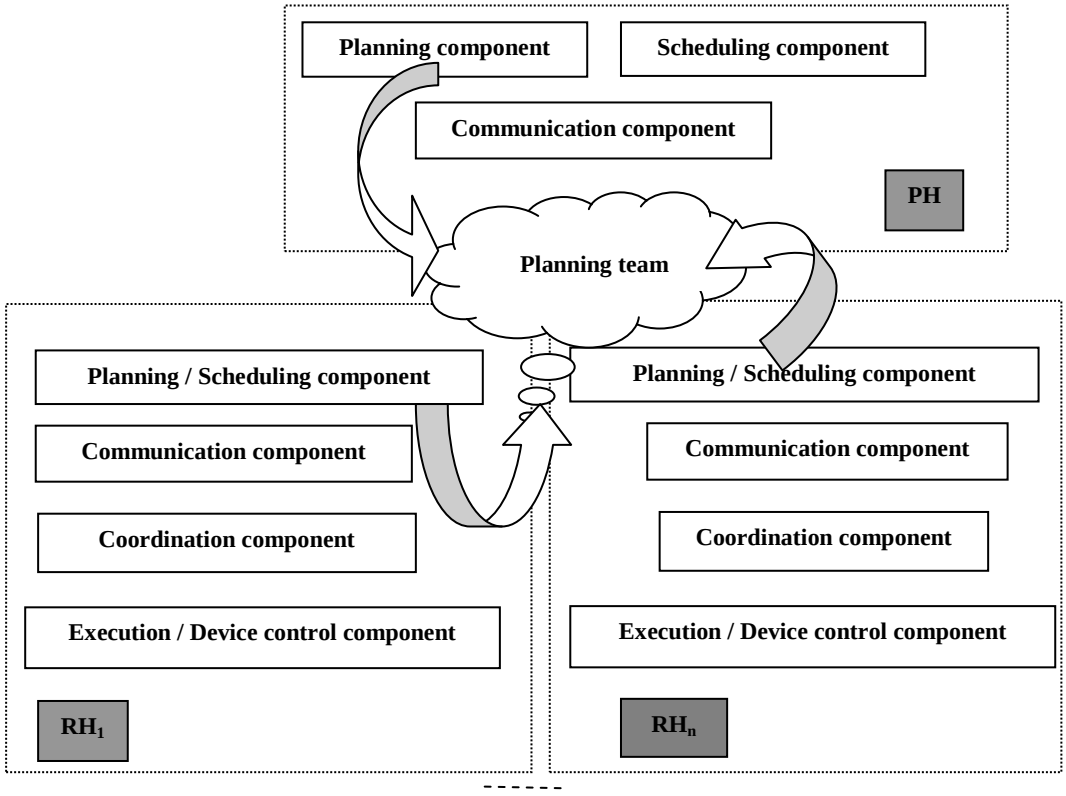

Fig. 2 HMS decentralization

Following the operation planning model, the other activities can determine a dynamic formation of collaborative teams that will assume both their own goals and the goals of the ongoing organization they belong to.

As it can be easily noticed, the holonic organization of the manufacturing modern processes involves decentralization mechanisms along with dynamic collaborative domains formation. Both characteristics define the system flexibility that will be the subject of the following paragraphs. As already sketched, in order to increase the system flexibility two aspects were taken into consideration: the decentralization of holons activities and the operational standardized aggregation of the resources in new defined cooperation domains.

3 Considering SOA for the HMS decentralized control

The HMS philosophy imposes a new architectural control design with a strong orientation to ad-hoc applications made up of loosely coupled functionality units, that are intrinsically

unassociated and have no calls to each other embedded in them. The already identified chunks of functionality will be implemented under a distributed multi-agent environment using a new software paradigm, namely the service oriented architecture (SOA) [1], [7]. The agent major characteristics will be the autonomy and self determination. The agent mobility is not so important for a manufacturing set-up, but sometimes it can be also taken into consideration. Each identified holon type of activity can be designed as a service and the overall control system will be implemented using a multi-agent wrapper model FIPA compliant for inter-agent communication at holon level and SOA guidelines for low-level middleware communication. This development model is able to offer appropriate rules for grouping of pieces of functionality named manufacturing services from the top of management level to the bottom production and distribution tasks along with supporting activities. The Open Group offers the following definition [9]: SOA is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform and standardized manner to offer, discover, interact and use capabilities to produce desired effects consistent with measurable preconditions and assumed expectations. In the opinion of the enterprise architects SOA will help business to adapt in a cost-effective manner to changing market conditions in the sense of promoting the reuse of services (functions at macro level) rather that the reuse of classes (functions of micro level). It will further simplify the interconnection with existing information technology legacy assets. Moreover, SOA promotes the separation of consumers from the service implementations that can be run on various different platforms and be accessed across networks.

The main goals of service oriented architecture and development is to construct complex software-intensive systems from a set of universally interconnected and interdependent blocks named services [1],[7] with the increasing of:

- The interoperability as information exchange and reusability;
- The federation as application of unifying mechanism while maintaining the autonomy and the self-governance;
- The alignment of business and technology domain.

The basic entity of any SOA is the service that comprises a stand-alone unit of functionality available only via a formally defined interface and that is implemented according to the principles of abstraction, autonomy, composability, discoverability, formal contract, loose coupling, reusability and statelessness. All these principles come in agreement with the concepts of holonic manufacturing architectures and the service-orientation seems to be the appropriate development strategy for HMS.

4 SOA implementing guidelines for HMS

Generally speaking, the holon should be viewed as a service able to appropriately combine both the SOA and HMS concepts. Thus, the holon is an entity which offers services and may use other holon services to accomplish its demanded goals. Making use of the SOA architecture, the Fig 3 sketches the grouping of the different holon types (order, product and resource holons) as an ad-hoc collection of holon services. The main propose of such a collection is to group stand-alone holon services into a service space where services are used/reused to carry out the proposed goals. The entire architecture is supported by the CORBA [5] standard which is a cross-platform approach with non-proprietary direction. This standard is used in HMS to resolve the integration problems of the physical devices in resource holons. It is adopted to the whole architecture in order to maintain a homogeneous approach. Moreover, the Foundation for Intelligent Physical Agents (FIPA) offers standards for the interoperation of heterogeneous software agent platforms, but still does not provide the means to integrate several physical resources in a common software structure. The technique to implement services using this standard is relied on the object invocation method through the *Bus-based Architecture* which includes the Internet Inter-Orb

Protocol (IIOP) communication layer. CORBA comes with a set of services that improves the communication layer and put forward the means for the holon service composition. Some CORBA services come near the holon services into Fig. 3, to supplement the scheme. Thus, the *Naming Service* manages a binding list between names and object references. It is used by each holon to register its services under friendly unique names and to locate the services by using the known names. To enable holons locate services based on the functionality and the quality of required services without knowing their names, the *Trader Service* is employed. Another useful service is the *Event Service,* which offers a mechanism for bidirectional service calls from the consumer holon to the service holon and in the reverse way. Besides these ones, others CORBA services [4] may be used, in order to solve specific issues from the HMS (e.g. service life cycle, concurrence control, relationships, etc.).

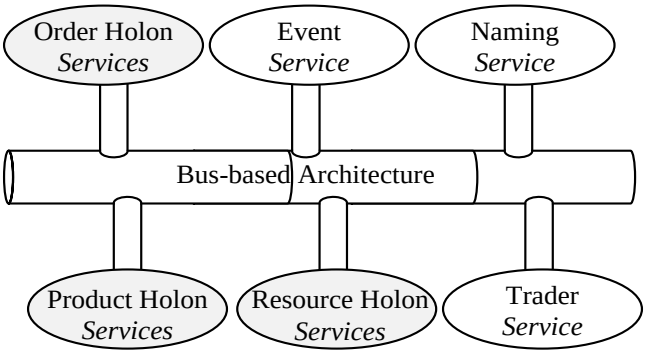

Fig. 3 SOA concept for HMS

From the implementation point of view, it is efficient that every holon should be designed to expose a single service, enabling the receiving of goal's description (Fig. 4). The service contains a remote object that encloses a single method to receive the content of the goal that describes the activity/activities that should perform by the service holon, and further methods (useful to administrate holon services). An Interface Description Language (IDL) file presents the scheme of the holon service that is the basis of the remote object interface; in CORBA, this object is referred as skeleton interface. The consumer of the holon service obtains the necessary object (known as the stub), for making a dynamic link with the reference of the remote object, from IDL description. The dynamic link between skeleton and stub can be established, if consumer service has the reference of the remote object (skeleton). Because the holon offers a service and may invoke other services, it knows the definition of both the skeleton and stub interfaces. The remote object is obtained from the implementation of the skeleton interface. Since there is a single service, this should deliver to the trader service at the activation time a list with the types of goals that can be carried out by the respective holon.

For example, in the case of a product holon the list can contain goal types like: stack or pallet products, or both product types. These lists are obtained at the time when holons publish the location and the reference of the remote objects and are used by the *Trader Service* to discover the appropriate holons for a specific goal. It maintains a repository with these remote objects and their associate proprieties that can be exposed if there is a request from other holons(Fig. 4). As an example, the order holon OH1 from Fig. 1 can request the addresses of the product holons PH Stack I or PH Stack II that can handle the stack product goal. If the order holon has received at least one remote address, it means there is a holon in the holonic system that may perform the stack assembling goal. Using these returned addresses, the holon can bind the stub objects to the object references. Through the service remote method invocation (the remote methods of the objects are invoked), the body of the goal message is transferred between holons. This communication model is made in a synchronous approach, which means that the client uses the

remote method and waits the server to respond. The use of the synchronous mode for SOA and HMS schemes is not enough, and the asynchronous mode is needed, too.

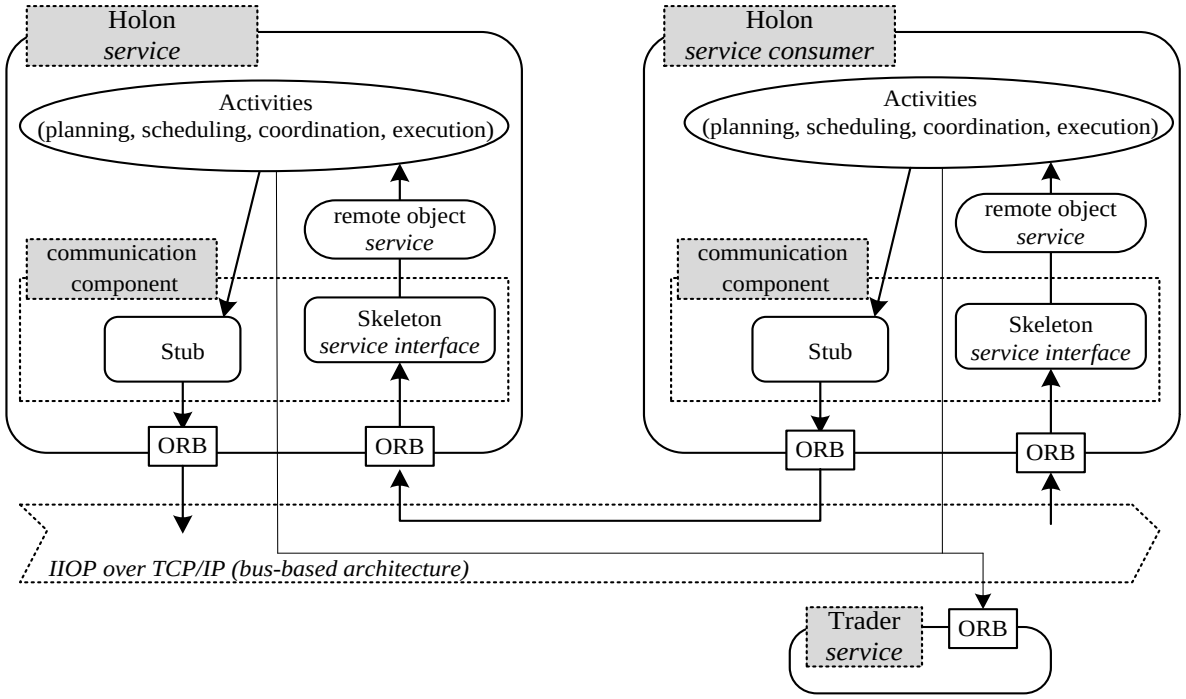

Fig. 4 The holon service model

For the asynchronous approach two mechanisms were used: through the remote method the message goal is transferred together with a remote reference object, which is used by the holon service to return an answer to the respective call; the other possibility is to use the *Event Service* to facilitate the bidirectional message exchange. In the first mechanism, the service' consumer needs to expose a remote object, able to handle an asynchronous response. Before the consumer forwards the goal, it activates the remote object which can be used by the holon service. Thus, the service knows the interface of the remote object (stub), which is identical for all the holons in the answering case. After receiving the goal message and an address, the service responses immediately and the customer should wait the appropriate answer in its active remote object at the given address. The second applicable mechanism that was used is more dynamic, and it is guided by the concept of subscribing and notification or the so called "change notification" [6]. Both holons (service and consumer) are using an event channel, which is a middleware between them, providing asynchronous communication of event data. For the communication process, three schemes (push-style, pull-style and mixed style schemes) can be used to involve the event channel [6]. As a service, the holon publishes a goal through the event channel, and this notifies all the consumer holons, which subscribed in the channel. In this way, the task to find holons that can fulfill a certain goal type is accomplished by the Event Service. Moreover, the consumer holons can pull the services through the channel to return an answer for the received event. Internally, when the holon receives a goal message through its remote object, the message is forwarded to the deliberative part to select a specific plan [8]. The chosen plan can either determines to carry out alone the received goal, or it becomes a consumer if the goal is divided in sub-goals. In the last case, the *Trader Service* is again used to discover holons for solving the sub-goals. The dynamic goal decomposition selects and uses the needed holon services from the service space. The solution and the performance of the completed goals depend on the dimension of the service space – the number of holons.

5 Conclusions

The purpose of the presented work is to sketch the decentralized control mechanisms existing in modern manufacturing systems and to identify software solutions in order to implement them. Starting from the holon entity, as recursive self organized unit and on the basis of hetararchical organization principles (the replacement of master-slave hierarchic control with collaboration relationships established after a negotiation process), the possible dynamic cooperation domains among different holon types in order to achieve a mutual goal/task were identified. In order to ensure the required flexible behavior of the control system, based on negotiation and real-time regrouping of active entities, the classical multi-agent implementation was improved by the service oriented architectural design vision. The paper proposes and gives some details on the correlation between SOA and HMS organizing concepts and proposes practical portable solutions. The authors have already obtained valuable results of the proposed architecture using the JACK ® Multi Agent Platform coupled with a CORBA based services implementation that will be the subject of further reports.

Acknowledgement: This paper is founded from the research project SOFHICOR, Contract no. 11-042/2007.

References

[1] M. Bell, *Introduction to Service-Oriented Modeling. Service-Oriented Modeling: Service Analysis, Design, and Architecture*. Wiley & Sons, 2008.
[2] V. Brussel, J. Wyns, P. Valckenaers, L. Bongaerts, P. Peeters, Reference architecture for holonic manufacturing systems: PROSA, *Computers in Industry*, vol. 37, pp. 255 – 274, 1998.
[3] M. Fletcher, T. Neligwa, chapter : An HMS Operational Model, from the volume *Agent Based Manufacturing,* editor S.M. Deen, Springer-Verlag, New York Heidelberg Berlin, pp. 163 – 192, ISBN:3-540-44069-0, 2003.
[4] T. Emmerich , *Engineering Distributed Object*, John Wiley&Sons, Baffins Lane, pp. 73-98, 190-200, 2000.
[5] OMG, *Common Object Request Broker Architecture Specification v3.1 part 1*, Technical report, Object Management Group, Inc, Needham, pp 13 – 28, 2008.
[6] OMG, *Event Service Specification v.1.2*, Technical report, Object Management Group, Inc, Needham, 2004.
[7] T. Erl, *Service-oriented Architecture: Concepts, Technology, and Design*. Upper Saddle River: Prentice Hall PTR, 2005.
[8] G. Varvara, Multi-agent Control System development for Manufacturing Processes based on Holonic Scenarios, *Proceedings of the RAAD 2009 18th International Workshop on Robotics in Alpe – Adria - Danube Region*, Brasov, Romania, May 2009.
[9] L.Wang, K. Chen, and Y.S. Ong (Eds.). An Agent-Based Holonic Architecture for Reconfigurable Manufacturing Systems. *ICNC 2005, LNCS 3612*, Springer-Verlag Berlin Heidelberg, pp. 622-627, 2005.
[10] A. Koestler, *The ghost in the machine*, in: Arkana Books, 1971.

GABRIELA VARVARA
"Gheorghe Asachi" Technical University
Faculty of Automatic Control and Computer Engineering
Department of Automatic Control and Applied Informatics
Mangeron 53A, 700050, Iaşi
ROMANIA
E-mail: gvarvara@ac.tuiasi.ro

CARLOS PASCAL
"Gheorghe Asachi" Technical University
Faculty of Automatic Control and Computer Engineering
Department of Automatic Control and Applied Informatics
Mangeron 53A, 700050, Iaşi
ROMANIA
E-mail: cpascal@ac.tuiasi.ro

DORU PĂNESCU
"Gheorghe Asachi" Technical University
Faculty of Automatic Control and Computer Engineering
Department of Automatic Control and Applied Informatics
Mangeron 53A, 700050, Iaşi
ROMANIA
E-mail: dorup@ac.tuiasi.ro