Abstract – Agent-based computing can be considered as a new general purpose paradigm for software development, which tends to radically influence the way a software system is conceived and developed, and which calls for new agent specific software engineering approaches. This paper presents an architecture for distributed manufacturing scheduling and follows Agent Oriented Software Engineering (AOSE) guidelines through specification defined by Ingenias methodology. This architecture is based on a Multi-Agent System (MAS) composed by a set of autonomous agents that cooperates in order to accomplish a good global solution.

I. INTRODUCTION

A major challenge in the area of global market economy is the development of new techniques for solving real world scheduling problems. Indeed, any industrial organization can only be economically feasible by maximizing customer services, maintaining efficient, low cost operations and minimizing total investment.

Traditional scheduling methods, encounter great difficulties when they are applied to some real-world situations. The interest in optimization algorithms for dynamic optimization problems is growing and a number of authors have proposed an even greater number of new approaches, and as a result, the field lacks a general understanding as to suitable benchmark problems, fair comparisons and measurement of algorithm quality [1][2][3][4].

Current practices and newly observed trends lead to the development of new ways of thinking, managing and organizing in enterprises, where autonomy, decentralization and distribution are some of the challenges. In manufacturing, a new class of software architectures, and organizational models appeared to give form to the Distributed Manufacturing System concept [5].

In the recent years, the characteristics and expectations of software systems have changed dramatically having as result that a variety of new software engineering challenges have arisen [6][7][8].

In this work we have two main purposes. Firstly, the resolution of more realistic scheduling problems in the domain of manufacturing environments, known as Extended Job-Shop Scheduling Problems [9][10], combining Multi-Agent Systems (MAS) and Meta-Heuristics technologies. Secondly, to demonstrate that is essential for MAS development the integration of Software Engineering concepts like the AOSE paradigm.

The proposed Team-based architecture is rather different from the ones found in the literature; as we try to implement a system where each agent (Machine Agent) is responsible to achieve a near optimal solution to schedule operations related with one specific machine through Tabu Search or Genetic Algorithms. After local solutions are found, each Machine Agent is required to cooperate with other Machine Agents in order to achieve a global optimal schedule.

The remaining sections are organized as follows: Section II summarizes some related work and the research on the use of multi-agent technology for dynamic scheduling resolution. In Section III are introduced some terms and definitions in Multi Agent Systems. This section presents some Agent-Oriented Methodologies and describes some considerations regarding Software Architectures and Multi-Agent Systems. In section IV the scheduling problem under consideration is defined. Section V presents the Team-Work based Model for Dynamic Manufacturing Scheduling and a proposal by Ingenias methodology.
Finally, the paper presents some conclusions and puts forward some ideas for future work.

II. RELATED WORK

Dynamic scheduling is one that is receiving increasing attention amongst both researchers and practitioners. In spite of all previous contributions the scheduling problem is still known to be NP-complete [2]. This fact incites researchers to explore new directions. Multi-Agent technology has been considered as an important approach for developing industrial distributed systems.

In [11] Shen and Norrie presented a state-of-the-art survey referencing a number of publications that attempted to solve distributed dynamic scheduling problems. According to these authors, there are two distinct approaches in the mentioned work. The first is based on an incremental search process that may involve backtracking. The second approach is based on systems in which an agent represents a single resource and is therefore responsible for scheduling that resource. Agents then negotiate with other agents in order to accomplish a feasible solution.

For further works developed on MAS for dynamic scheduling, see for example, [9][12].

The characteristics and expectations of software systems have changed dramatically in the last few years, with the result that a range of new software engineering challenges have arisen [6][7]. First, most software systems are concurrent and distributed, and are expected to interact with components and exploit services that are dynamically found in the network. Second, software systems are becoming “always-on” entities that cannot be stopped, restored, and maintained in the traditional way. Finally, current software systems tend to be open, because they exist in a dynamic operating environment where new components can join and existing components can leave the system on a continuous basis, and where the operating conditions themselves are likely to change in unpredictable ways.

From the literature we can conclude that Agent-based computing is a promising research approach for developing applications in complex domains. However, despite the great research effort [8][13][14], there still exists a number of challenges before making agent-based computing a widely accepted paradigm in software engineering practice. In order to realize an engineering change in agent oriented software, it is necessary to turn agent oriented software abstractions into practical tools for facing the complexity of modern and current application areas.

III. MULTI-AGENT SYSTEMS

Agents and multi-agent systems (MAS) have recently emerged as a powerful technology to deal with the complexity of current Information and Communication Technologies environments. In this section we will describe some issues and considerations regarding the developing of the MAS following a software engineering perspective.

A. Terms and Definitions

The development of multi-agent systems requires powerful and effective modelling, architectures, methodologies, notation techniques, languages and frameworks. Agent-based computing can be considered as a new general purpose paradigm for software development, which tends to radically influence the way a software system is conceived and developed, and which calls for new, agent specific software engineering approaches [8].

The main term of Multi-Agent based computing is an Agent. However the definition of the term Agent has not common consent. In the last few years most authors agreed that this definition depends on the domain where agents are used. In Ferber [15] is proposed a definition: “An agent is a virtual or physical autonomous entity which performs a given task using information gleaned from its environment to act in a suitable manner so as to complete the task successfully. The agent should be able to adapt itself based on changes occurring in its environment, so that a change in circumstances will still yield the intended result.”

An agent can be generally viewed as a software entity with characteristics [16] such as:

- Autonomy - where an agent has its own internal thread of execution, typically oriented to the achievement of a specific task, and it decides for itself what actions it should perform at what time.
- Situatedness - agents perform their actions while situated in a particular environment.
- Proactivity - in order to accomplish its design objectives in a dynamic and unpredictable environment the agent may need to act to ensure that its set goals are achieved and that new goals are opportunistically pursued whenever appropriate.
- Sociability - agents interact (cooperate, coordinate or negotiate) with one another, either to achieve a common objective or because this is necessary for them to achieve their own objectives.

A Multi-Agent System (MAS) can be defined as “a system composed by a population of autonomous agents, which cooperate with each other to reach common objectives, while simultaneously each agent pursues individual objectives” [15]. According to Russell and Norving [17] multi-agent systems “[…] solve complex problems in a distributed fashion without the need for each agent to know about the whole problem being solved”.

We can see MAS like a society of agents that cooperates to work in the best way possible. With this we gain the ability of solve complex problems like dynamic and distributed scheduling. Considering the complexity inherent to the manufacturing systems, the