Forschungsbericht
Institut für Automatisierungs- und Softwaretechnik

Hrsg.: Prof. Dr.-Ing. Dr. h. c. P. Göhner

Iman Badr

Agent-Based Dynamic Scheduling for Flexible Manufacturing Systems

Band 1/2011

Universität Stuttgart

IAS-Forschungsberichte

Agent-Based Dynamic Scheduling for Flexible Manufacturing Systems

Von der Fakultät Informatik, Elektrotechnik und Informationstechnik
der Universität Stuttgart zur Erlangung der Würde eines
Doktor-Ingenieurs (Dr.-Ing.) genehmigte Abhandlung

Vorgelegt von
Iman Asaad Badr
aus Kairo, Ägypten

Hauptberichter:	Prof. Dr.-Ing. Dr. h. c. Peter Göhner
Mitberichter:	Prof. Dr.-Ing. Alexander Fay
Tag der Einreichung:	09.06.2010
Tag der mündlichen Prüfung:	23.11.2010

Institut für Automatisierungs- und Softwaretechnik
der Universität Stuttgart

2010

IAS-Forschungsberichte

Band 1/2011

Iman Badr

Agent-Based Dynamic Scheduling for Flexible Manufacturing Systems

D 93 (Diss. Universität Stuttgart)

Shaker Verlag
Aachen 2011

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Zugl.: Stuttgart, Univ., Diss., 2010

Printed in Germany.

ISBN 978-3-8322-9736-7
ISSN 1610-4781

Shaker Verlag GmbH • P.O. BOX 101818 • D-52018 Aachen
Phone: 0049/2407/9596-0 • Telefax: 0049/2407/9596-9
Internet: www.shaker.de • e-mail: info@shaker.de

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftliche Mitarbeiterin am Institut für Automatisierungs- und Softwaretechnik (IAS) der Universität Stuttgart.

Zuallererst geht mein Dank an Allah, den Allmächtigen, der mir das Wissen, das Selbstvertrauen und die Kraft gegeben hat, die vorliegende Arbeit zu vollenden.

Mein besonderer Dank gilt meinem Doktorvater und Leiter des Instituts, Herrn Prof. Dr.-Ing. Dr. h. c. Peter Göhner, für die fortwährende Unterstützung während der Entstehung der Arbeit, die zahlreichen wertvollen Anregungen und kritischen Diskussionen sowie die Übernahme des Hauptberichts.

Herrn Prof. Dr.-Ing. Alexander Fay danke ich für das Interesse an meiner Arbeit und die Übernahme des Mitberichts.

Allen ehemaligen Kolleginnen und Kollegen am IAS gilt mein Dank für die gute Zusammenarbeit und die hilfsbereite Unterstützung. Ganz besonders bedanke ich mich bei Michael Wedel und Hisham Mubarak für die regelmäßigen hilfreichen Diskussionen über mein Forschungsthema sowie die kritische und gründliche Durchsicht des Manuskripts. Ebenfalls bin ich Stephan Pech dankbar für die Mitwirkung bei der Durchsicht des Manuskripts und für die stetige Hilfsbereitschaft.

Erwähnen möchte ich auch die Studierenden, die im Rahmen ihrer Diplom- und Studienarbeiten einen großen Beitrag zum Gelingen dieser Arbeit geleistet haben.

Der ägyptischen Regierung danke ich für die finanzielle Unterstützung, die diese Arbeit ermöglicht hat.

Schließlich möchte ich mich ganz herzlich bei meiner Familie bedanken, insbesondere bei meinem verstorbenen Vater für die fortwährende Förderung und bei meiner Mutter für die grenzenlose Unterstützung und Fürsorge.

Stuttgart, im Dezember 2010

Iman Badr

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AGV	**A**utomatic **G**uided **V**ehicle
CAD	**C**omputer **A**ided **D**esign
CFP	**C**all **f**or **P**roposal
CIM	**C**omputer **I**ntegrated **M**anufacturing
CNC	**C**omputer **N**umerically **C**ontrol
CNET	**C**ontract **Net**
DAO	**D**ata **A**ccess **O**bject
DTD	**D**ocument **T**ype **D**efinition
EDD	**E**arliest **D**ue **D**ate
FCFS	**F**irst **C**ome **F**irst **S**erved
FMC	**F**lexible **M**anufacturing **C**ell
FMS	**F**lexible **M**anufacturing **S**ystem
GA	**G**enetic **A**lgorithm
GADWAL	Inte**g**rated **A**gent-Base**d** Frame**w**ork for Dyn**a**mic Schedu**l**ing
GT	**G**roup **T**echnology
GUI	**G**raphical **U**ser **I**nterface
IAS	Institute of Industrial Automation and Software Engineering of Stuttgart, from its original German name “**I**nstitut für **A**utomatisierungs- und **S**oftwaretechnik”
JSSP	**J**ob **S**hop **S**cheduling **P**roblem
LS/TS	**L**iving **S**ystems / **T**echnology **S**uite
NA	**N**ot **A**pplicable
PPC	**P**roduction **P**lanning and **C**ontrol
SPT	**S**hortest **P**rocessing **T**ime
s.t.	**S**uch **t**hat
XML	E**x**tensible **M**arkup **L**anguage

Glossary

Agent: An encapsulated software entity with predefined goals. An agent endeavors to achieve its goals through its autonomous behavior and in doing so it interacts with its environment and with other agents.

Allocation: A sub-problem of the manufacturing scheduling problem corresponding to the assignment of resources to jobs to perform a given operation.

Efficiency: The quality of scheduling that judges the extent to which the resulting schedules are optimizing a given set of criteria.

Flexible Manufacturing Systems: Manufacturing systems consisting of computer numerically controlled machines interconnected by an automated material handling system, usually applying automatic guided vehicles, and controlled by an integrated computer system.

Framework: A generic software solution for a specific application area, where the flow of control is specified but the software entities can be extended, reused and configured to fit with the specific application for which the framework is to be used.

Job: A manufacturing task corresponding to the production of a certain part type with a certain quantity, where possibly a release time and a deadline are specified.

Manufacturing Flexibility: The flexibility of a system corresponds to its ability to adapt to its changing environment. In the context of manufacturing, flexibility is associated with reconfiguring manufacturing resources to react efficiently to the request of producing different products of acceptable quality [SeSe90].

Manufacturing Scheduling: The art of assigning resources to jobs over time, in a way that optimizes given criteria.

Optimization Problem: A problem of searching for a feasible solution that maximizes or minimizes a given set of criteria.

Machine Flexibility: A property of manufacturing systems resulting from having machines that are capable of supporting different operations and different part types without having to redesign the machines.

Makespan: The difference between the earliest start time and the latest end time for a set of scheduled jobs.

Pathfinding: The task of searching for the optimal path from a given source to a predefined destination

Production Scheduling: Allocating machines to jobs over time, in a way that optimizes a given set of criteria.

Rescheduling: The process of continuously updating a generated schedule in reaction to dynamic events necessitating changes to the schedule.

Routing: The sub-problem of transportation scheduling concerned with finding the best path for transporting workpieces from a given source to a given destination. Both the source and the destination can be the local buffer of a machine or a store of workpieces.

Routing Flexibility: The capability of manufacturing systems to produce the same part by assigning it to different machines, where the transportation of workpieces between these machines can take different routes.

Sequencing: A sub-problem of the production scheduling problem which is concerned with assigning time slots to an allocated set of jobs at a given machine.

Shop Floor: The space in the factory consisting of the processing resources and the material handling resources.

Transportation Scheduling: Planning the provision of workpieces to the required machines over time in such a way that satisfies given criteria such as the minimization of the transportation time.

Zusammenfassung

Die Notwendigkeit, im Markt auf Schwankungen und Vielfalt der Nachfrage sowie auf den harten Wettbewerb zu reagieren, hat zu dem zunehmenden Trend geführt, eine Vielzahl von Produkttypen in Kleinserien zu produzieren. Die Einführung von fortschrittlichen Technologien, wie numerisch gesteuerte Maschinen und fahrerlose Fahrzeuge, hat die Realisierung von flexiblen Fertigungssystemen (FFS) ermöglicht. Ziel dieser flexiblen Fertigungssysteme ist es, die in der Massenproduktion erreichte Effizienz auf die kleine bis mittlere Serienfertigung mit hoher Produktvielfalt zu übertragen. Dieses Ziel setzt Ablaufplanungsansätze voraus, die die eingesetzte Technologie effizient ausnutzen und zugleich eine gewisse Flexibilität in der Reaktion auf die Dynamik im Produktionsumfeld aufweisen. Konventionelle Ansätze der Ablaufplanung sind nicht in der Lage, eine rechtzeitige Reaktion der FFS auf die Dynamik im Produktionsumfeld zu gewährleisten. Ansätze, die eine sorgfältige Untersuchung der verfügbaren alternativen Ablaufpläne verfolgen, sind auf Grund der hohen Komplexität des Problems nicht geeignet, da sie Echtzeit-Bedingungen verletzen.

In dieser Arbeit wurde ein agentenbasiertes Konzept für die flexible und effiziente FFS-Ablaufplanung entwickelt. Die Reduzierung der inhärenten Komplexität des FFS-Ablaufplanungs-Problems wird durch die Zerlegung in autonome Agenten erreicht. Diese Agenten sind in einer mehrschichtigen Architektur, die auf der Flexibilität der FFS aufbaut, hierarchisch angeordnet. In jedem der beteiligten Agenten sind Suchheuristiken realisiert, die es ihm ermöglichen, die ihm gestellten Aufgaben aus seiner lokalen Perspektive zu optimieren. Durch die Interaktion zwischen den Agenten auf den verschiedenen Abstraktionsebenen wird der Ablaufplan aus der globalen Perspektive in angemessener Zeit optimiert. Um die verschiedenen Führungsentscheidungen und die verschiedenen Umgebungsbedingungen zu berücksichtigen, werden verschiedene Ablaufplanungsmodi unterstützt. Der generierte Ablaufplan wird anhand einer Methode, die für eine automatische Reaktion auf Störungen effizient und in Echtzeit sorgt, an Störeinflüsse wie Maschinenausfälle angepasst. Darüber hinaus können strukturelle Veränderungen der FFS, einschließlich der Hinzufügung neuer Ressourcen, dynamisch in die vorgeschlagene Zeitplanung, die zur Gewährleistung einer langfristigen Flexibilität dient, übernommen werden.

Abstract

The need to react to fluctuations and versatility in market demand and face competition threats has led to the increasing trend to produce a wide variety of product types in small batches. The advent of advanced technology like computerized numerically controlled (CNC) machines and automatic guided vehicles has enabled the realization of flexible manufacturing systems (FMSs). FMSs aim at bringing the efficiency of mass production to small-to-medium sized batch production with high product diversity. This objective calls for scheduling approaches that optimize the utilization of the applied technology and at the same time react to the environmental dynamics flexibly. Conventional scheduling approaches fail to provide a mechanism for reacting to the dynamics of FMSs in a timely and efficient manner. Approaches that cater for optimality by a thorough investigation of available schedule alternatives always fail to exhibit real-time reactivity due to the high complexity of the problem.

In this research work, an agent-based concept for the flexible and efficient FMS scheduling is proposed. The inherent complexity of the FMS scheduling problem is tackled by decomposing it into autonomous agents. These agents are organized in a heterarchical multi-layered architecture that builds on the flexibility of FMSs. Every involved agent applies search heuristics to optimize its assigned task out of its local perspective. Through the interactions among the concerned agents along the different levels of abstraction, the schedule is optimized from the global perspective in reasonable time. Different scheduling modes are supported to account for the different managerial decisions and the different environmental conditions. The generated schedule is adapted to disturbing events such as machine breakdowns based on a schedule repair method that caters for automating the reaction to disturbances efficiently in real-time. In addition, structural changes of FMSs, including the addition of new resources, are incorporated dynamically into the proposed scheduling, which guarantees long-term flexibility.

1 Introduction

1.1 Background and Motivation

The beginning of the twentieth century has witnessed a revolution in the automotive industry in particular and in the manufacturing industry in general. For the first time in history, automobiles became affordable for the middle-class customers and turned into mass products. Through his innovative assembly line, Henry Ford, the famous car manufacturer, managed to radically reduce the assembly time and thus the production costs of the model T car, for which this line was dedicated. Model T automobiles were massively produced and dominated the market for decades due to their low prices. As other automotive manufacturers started to provide more variations in styling and design with competitive prices, Ford started to lose market share as a result of the rigidity of the assembly lines at that time, which represented an obstacle in satisfying the changing customer needs [Wiki10]. It turned out that the ability of manufacturing systems to adapt to the changing customer needs has become essential for surviving the rising competition.

As a result of economic globalization, the market is currently featuring growing competition manifested in the increasing trend towards diversifying the supported products and shortening their lifetime. The ability of manufacturers to survive in this highly competitive market is largely determined by their flexibility to adapt to changing customer needs. More concretely, the company that gains the lead is usually the one which reads the market changes and reacts to them before the others. The need to deal with market uncertainties and to quickly react to changes has motivated researchers and practitioners to envision the construction of flexible manufacturing systems (FMSs). Instead of designing products a priori and constructing devoted rigid manufacturing lines that are specifically intended for these products, the underlying concept of flexible manufacturing systems is to employ manufacturing resources that can be easily configured to produce wide spectrum of products and allow for the changeover of the production output with reduced effort.

The advent of advanced technology like universal machines, that support a variety of operations and product types, has caused flexible manufacturing systems to come into existence. By employing a limited number of flexible resources, flexible manufacturing systems aim at bringing the efficiency of mass production to small-to-medium sized batch production with high product diversity. They are associated with advantages like inventory reduction, decreased lead and delivery times, increased flexibility and improved quality [RaTc90] [Upto92]. From the time of their emergence, FMSs continued to diffuse and replace conventional manufacturing systems in several manufacturing industries, such as the automotive industry, the

semiconductor industry and the steel industry, to name a few [Wall03] [Witt90] [JLMF07]. The emergence of FMSs has posed challenges to the traditional manufacturing control functionalities in general and to scheduling in particular.

1.2 Challenge of Scheduling for Flexible Manufacturing Systems (FMSs)

Manufacturing scheduling addresses the allocation of available resources to requested jobs in a way that optimizes given criteria, like the time taken to finish all jobs. Conventionally, the scheduling problem was concerned with dedicated resources and statically defined jobs. Jobs used to be defined over a fixed time period according to the available capacity of the system resources and the current material supply [Grav81]. Accordingly, the input to scheduling was relatively deterministic and static, which permitted performing it isolated from other interrelated functionalities like shop floor control, which is concerned with the schedule execution. This motivated researchers to conceive manufacturing scheduling as an optimization problem and deal with it in isolation from its environment, which is usually assumed to be deterministic and static.

The advent of FMSs has brought about the challenge of achieving the required optimality while exhibiting flexibility to unforeseen events. A solution that caters for optimality by a thorough investigation of the available alternatives always fails to exhibit real-time reactivity due to the high complexity of the schedule optimization problem, and is thus regarded as inflexible. The adoption of a static scheduling approach for FMSs in practice is only possible through manual interventions based on ad hoc methods, which lowers the overall performance of FMSs and underutilizes their capacity as well as their potential for being operated with automated control [Herm06] [VHL03]. This underutilization of FMSs, mainly resulting from the limitations of the adopted scheduling approaches, led some researchers to question the economics of investing in these advanced technology and to argue that in most of the cases investments in FMSs do not pay back [RaTc90].

1.3 Objective of the Research Work

The objective of this research work is to develop a concept for scheduling that capitalizes on the flexibility of FMSs in adapting the production flow to the environmental dynamics in an efficient way. Towards achieving this objective, the following lower-level objectives have to be accomplished:

Maximizing the utilization of the available resources

Maximizing the utilization of the available resources is a classical objective of manufacturing scheduling, imposed by the need to increase the level of productivity. This objective gains more

significance when it comes to FMS scheduling. This is basically due to the limited number of resources employed in FMSs compared to other conventional manufacturing systems coupled with the high investment associated with these highly advanced resources. The lack of concepts accomplishing this objective has contributed to the underutilization of the FMSs, installed world wide [RaTc90]. For this investment to pay back, the production flow has to be scheduled in a way that maximizes the utilization of these resources by exploiting the available flexibility.

Applicability in practice

The concept to be developed has to account for practical considerations in typical FMSs and deal with the scheduling problem within its environment and not isolated from it. Scheduling has thus to be performed dynamically in a way that automates the reaction to potential changes in the operating environment. In other words, events affecting the schedule during operation have to be handled with no or minimal human intervention. In addition, changes to the FMS itself intended to provide better support to new customer needs, by for example adding a new resource, should be incorporated in the scheduling system, with minimum effort.

By fulfilling the two objectives, the desired combination of the optimality of the generated schedules with the flexibility of the scheduling system will be attained. While the optimality of the generated schedules are addressed by the first objective, the flexibility of the scheduling system is covered by the second objective.

1.4 Organization of the Thesis

The thesis consists of eleven chapters that are structured as follows.

Chapter 2 presents the fundamentals of FMS scheduling. It consists of two main parts. In the first part, flexible manufacturing systems are explored. For gaining good understanding of FMSs, the concept of manufacturing flexibility is first reviewed by defining and classifying the different types of flexibilities. FMSs are then defined and their structure is explained. The second part of this chapter presents the fundamentals of FMS scheduling. It defines the scheduling environment as well as the scheduling problem itself. The FMS scheduling problem is defined both as an optimization problem and as an iterative continuous process. The chapter concludes by deriving the requirements on the concept.

Chapter 3 surveys the conventional scheduling approaches out of a practical point of view. Due to the complexity of FMS scheduling and the lack of theoretical approaches that provide a comprehensive support for it, the evaluation considers the approaches from three different perspectives: the scheduling scope, the system architecture and the scheduling technique. Out of every perspective, a classification of the approaches is presented and assessed by referring to the requirements derived in the previous chapter. The chapter is concluded with a summary of the evaluation results and an outlook on the fundamental decisions for the proposed concept.

Chapter 4 introduces to the proposed concept by discussing the fundamental decisions taken to combine the flexibility of the scheduling system with the efficiency of the generated schedules. First, the decision of adopting an agent-based approach for attaining the desired flexibility is explained. In doing so, the advantages of agents in fulfilling the stated requirements are reasoned about – after briefly introducing to the underlying concept of agency. Second, the application of search heuristics for fulfilling the efficiency requirement is explained by presenting the basics of the selected algorithms and highlighting their suitability to scheduling. Finally, an overview on the supported scheduling modes is presented.

The proposed concept is addressed in chapters 5 - 8. Chapter 5 embarks on the proposed system architecture. First, a model for the surrounding environment is developed and the necessary interfaces connecting it with the system are described. Second, the multi-agent architecture is derived and the necessary agent types are identified. Finally, the internal agent architecture is explained and illustrated.

Chapter 6 addresses the inter-agent communications required for schedule generation. It starts with an illustration of the different levels of cooperation among agents. Following this illustration, it gives an insight into the interplay among agents by explaining the communication protocols for schedule generation under consideration of the different schedule generation modes and the different cooperation levels.

Chapter 7 discusses the schedule generation process. The behavior of the agent types participating in the schedule generation process is explained. The explanation of the behavior focuses on details concerning the applied scheduling techniques. Details concerning the schedule generation under the different configurations and environmental conditions are also described.

Chapter 8 is devoted to schedule repair, i.e. the adaptation of the generated schedule to unforeseen events causing disturbances to it. It starts with analyzing possible disturbances to the schedule by developing a disturbance model which is used in presenting taxonomy of disturbances. Following this analysis of disturbances, a method for disturbance handling is proposed. This method is specially devised for adapting schedules with distributed structures, such as the case in the proposed agent-based scheduling. Details for extending the agents with this method are then presented. Finally, the applicability of the presented extension in handling all potential disturbances, analyzed in the beginning of the chapter, is investigated.

Chapter 9 presents the details of realizing the proposed concept based on a generic scheduling framework that can be deployed to simulated FMSs, and is envisioned to be tested on real FMSs. Use cases of this framework are illustrated. Details of the implementation of the agents and their integration within the environment are explained. Finally, details for modeling FMSs, configuring their simulation models and running them are given.

Chapter 10 evaluates the proposed concept using an FMS model. It gives first an overview on the considered system. The evaluation is based on results attained from experiments, that aim at assessing the performance of the different supported scheduling modes. The configuration of these experiments is guided by the stated requirements. In light of the illustrated experimental results, concluding remarks about the fulfillment of the proposed concept to the stated requirements are derived.

Chapter 11 concludes the thesis by highlighting the significant characteristics of the presented concept and pointing out the main contributions of the research work. In addition, possible research directions for future work are presented.

2 Fundamentals of Scheduling for Flexible Manufacturing Systems

This chapter lays the foundations of scheduling for FMSs. It is divided into three main sections. Section 2.1 reviews the fundamental concepts of FMSs. It starts by addressing the concept of flexibility in manufacturing as a main factor in conceiving the fundamentals of FMSs. Following the illustration of flexibility, FMSs are defined and their internal structure is explored. In light of this discussion, FMS scheduling is addressed in section 2.2. The FMS scheduling problem is defined after briefly investigating the scheduling environment. Tackling the FMS scheduling problem is explained from two perspectives: by regarding it as an optimization problem and as an iterative dynamic process. While the former accounts for the theoretical viewpoint the latter is concerned with the practical standpoint. The chapter concludes with section 2.3, which derives the requirements on FMS scheduling.

2.1 Flexible Manufacturing Systems

The concept of FMSs is credited to David Williamson, a British engineer, who received a patent in 1965 for his invention of the FMSs concept. One of the earliest installations of FMSs took place in the Federal Republic of Germany in 1969 at Heidelberger Druckmaschinen in cooperation with the University of Stuttgart [Groo08]. This concept has gained growing interest and applicability from academia and industry, which is reflected in the number of FMS installations worldwide. According to [WeHy87] [RaTc90], the number of installed FMSs world wide was raised from 80 in the beginning of the 80's to around 1200 installations by the 90's and was estimated to reach 3500 by the year 2000 [Shiv06].

Flexible manufacturing systems aim at combining the efficiency of fixed automation like transfer lines with the flexibility[1] of programmable automation such as standalone computer numerically control (CNC) machines. In other words, they enable bringing the efficiency of mass production to small-to-medium sized batch production with high product diversity. As illustrated in Figure 2.1, the parts made by FMSs are of medium volumes and moderate diversity. The appropriate production volume range is 5,000 to 75,000 parts per year. If the annual production is below this range, an FMS is likely to be an expensive alternative. If production volume is above this range, then a more specialized production system should probably be considered [Groo08].

[1] In this context, flexibility refers to the degree of variations in part types that can be supported by the manufacturing system.

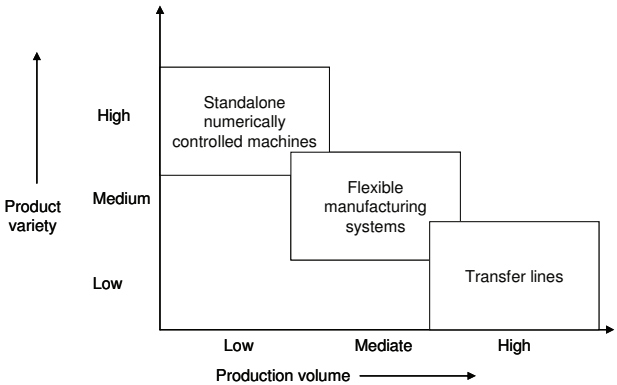


Figure 2.1: Production volume and diversity of FMSs [Shiv06]

The main distinguishing feature shared by all flexible manufacturing systems is flexibility. Therefore, a precise definition of FMSs requires embarking on the concept of flexibility in manufacturing.

2.1.1 Flexibility in Manufacturing

The flexibility of a system corresponds to its ability to adapt to its changing environment. In the context of manufacturing, flexibility is associated with reconfiguring manufacturing resources so as to produce efficiently different products of acceptable quality. The concept of flexibility is dealt with from a multitude of perspectives. In general, flexibility has been considered from the economic and the organizational views. From an economic view, flexibility corresponds to the extent to which the average cost varies in reaction to environmental changes. In other words, the lower the costs incurred by a factory to adapt to changing demand, the higher its flexibility would be. On the other hand, flexibility is viewed from an organizational perspective as the ability of an organization to suffer limited change without severe disorganization. This view is concerned with the internal resources and the extent to which it can accommodate for environmental changes with little or no internal changes [SeSe90].

More specifically, flexibility can be viewed as a multidimensional concept comprising a variety of types [SeSe90]. In what follows, a classification of the main types of flexibility, illustrated in Figure 2.2, is reviewed.

2.1.1.1 Component or Basic Flexibilities

This class of flexibilities is related to the individual components constituting the whole system. There are basically three types belonging to this category [SeSe90].

- Machine flexibility corresponds to the diversity of operations supported by a machine without prohibitive effort. The higher the number of operations supported by a machine, the higher its flexibility is. Operations include drilling, grinding, assembling, etc. Stating that the change has to be without prohibitive effort implies that switching the operation on a certain machine can be done without having to redesign the machine. This is currently enabled by advanced technology including numerical control, sophisticated part loading and tool changing.

- Material handling flexibility refers to the ability of loading, unloading and moving different part types throughout the manufacturing facility efficiently. Having this type of flexibility increases the utilization of machines. It is attained by means of flexible layout design and technology like automatic guided vehicles (AGVs) and general-purpose fixtures.

- Operation flexibility is attributed to the manufactured parts and represents the ability of a certain part to be produced in different ways or with alternate process plans. A process plan corresponds to the sequence of operations required to manufacture a given part. Operation flexibility is controlled during the design of the parts and results in having more alternatives for allocation, which implies higher flexibility in scheduling.

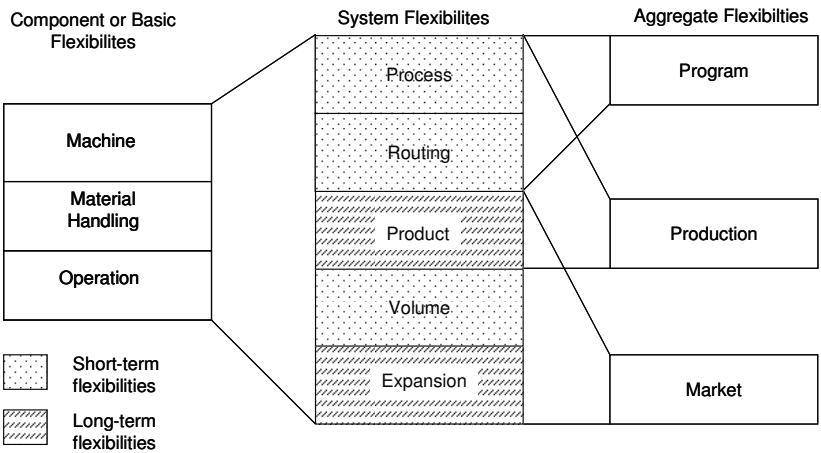


Figure 2.2: Associating the various types of flexibilities – adapted from [SeSe90]

2.1.1.2 System Flexibilities

As illustrated in Figure 2.2, there are mainly five types of flexibilities that a manufacturing system should exhibit [SeSe90].

- Process flexibility refers to the mix flexibility or the set of part types that the system can produce without major setups. It corresponds to the ability of the system to meet the diversity in customer demand by means of the simultaneous manufacturing of a variety of product lines. This variation in part types is usually managed based on the group technology (GT) which is a manufacturing philosophy based on grouping to capitalize on similarities both in manufactured parts and in machines [WeHy87].

- Routing flexibility corresponds to the ability of the system to produce the same part with alternative routes through the system. It results from an efficient balancing of the current load while scheduling tasks. It implies having several feasible routes at a given point in time for producing a certain part, which serves in dealing efficiently with unanticipated events like machine breakdowns.

- Product flexibility relates to the ease of applying changes to planned products by adding new parts or substituting existing ones. Product flexibility is sometimes termed as modification flexibility. It is regarded as essential in facing competition especially in markets with short or uncertain product lifecycles. It can be measured based on the time or cost required to change the existing product mix.

- Volume flexibility refers to the ability to change the planned output level. It helps in facing uncertainty and fluctuations in demand. It is typically measured in terms of the range of volumes with which the system can run profitably. A related type of flexibility is delivery flexibility which represents the ability to meet changes in planned delivery dates.

- Expansion flexibility is concerned with increasing the capacity and capability of the system by adding or replacing machinery. It is usually evaluated in terms of the overall time needed to achieve a given capacity level.

As illustrated in Figure 2.2, while process, routing and volume flexibilities are referred to as short-term flexibilities, product and expansion flexibilities correspond to the long-term flexibility.

2.1.1.3 Aggregate Flexibilities

Aggregate flexibilities result from exhibiting certain system flexibilities. As illustrated in Figure 2.2, three main aggregate flexibilities can be identified [SeSe90].

- Program flexibility represents the ability of the system to run almost unattended, which corresponds to the degree of automation satisfied by the computer control. Researches usually require the system to run unattended during the second and the third shifts of operation. This is due to the inevitable need to perform some manual operations like inspection, fixturing and maintenance, which can be scheduled in the first shift. Program flexibility is achieved by means of process and routing flexibilities. In addition, it is based

on the existence of sensors for reporting unanticipated events and computer control enabling a suitable automated reaction.

- Production flexibility is defined as the set of part types that the system can produce without the addition of major capital equipment. This definition allows for adding minor equipments like cutting tools without degrading production flexibility. It is worth noting that while product flexibility is associated with the system reaction to a certain change in the planned products, production flexibility is concerned with the range of part types produced on the long run. It is an aggregate of process flexibility, routing flexibility and product flexibility.
- Market flexibility is associated with the ease with which a system can adapt to changes in the market environment. It represents a collective term for product, volume and expansion flexibilities. It is essential in case of customized products and fluctuating demands.

2.1.2 Definition of FMS

In spite of the lack of a standard definition for flexible manufacturing systems, the term FMS is usually associated with flexibility and automation [SeSe92] [RaSt94]. In what follows, three of the most frequently referenced definitions of FMSs are listed:

- An integrated computer-controlled complex of automated material handling devices and numerically controlled machine tools that can simultaneously process medium-sized volumes of a variety of part types [Kouv92].
- A highly automated group technology machine cell, consisting of a group of processing stations (usually CNC machine tools), interconnected by an automated material handling and storage system, and controlled by an integrated computer system [Groo00].
- A production system consisting of a set of identical and/or complementary numerically controlled machines which are connected through an automated transportation system. Each process in an FMS is controlled by a dedicated computer (FMS cell computer). This is often embedded in a large hierarchical network of computers [TeKu93].

It is evident from these definitions that the concept of flexible manufacturing systems is associated with the employed technology and the grouping principle referred to as the group technology. It is essential to have computer-controlled machines connected with an automated material handling and transportation system and managed by computer control. Necessitating these primary components as well as employing group technology ensures a certain degree of component or basic flexibility, as illustrated in section 2.1.1.1.

2.1.3 Structure of FMS

As industrial automation systems, FMSs consist of a technical system, an information system and a human operator system, as illustrated in Figure 2.3 [TaHa88] [TeKu93]. The technical system of FMSs comprises typically a processing system supported by a workpiece supply system and a tool supply system in addition to supplementary systems like energy and waste disposal systems.

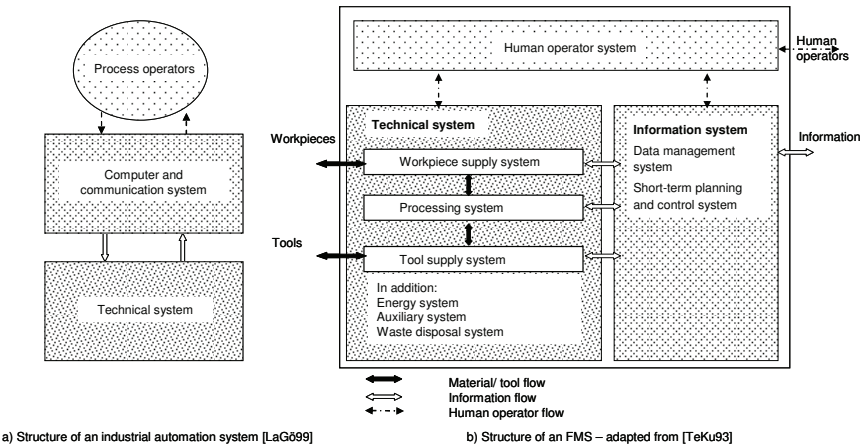


Figure 2.3: Subsystems of an FMS with the corresponding structure of a general industrial automation system

The processing system is composed mainly of a set of machines with local tool exchange and storage systems. The machines are typically numerically controlled and can be classified according to their capabilities into universal machines and specialized machines. Universal machines are capable of performing several cutting operations like drilling milling and turning based on tool exchange – a feature which is not supported by specialized machines. In FMSs, most of the employed machines are numerically controlled universal machines.

The workpiece supply system consists of a transport system, a storage system, a setup system, and a handling system. The transport system controls the flow of the workpieces among the individual machines, the storage system and the setup system. A typical transport system of an FMS is dominated by automatic guided vehicles. The workpiece storage system is responsible for storing the workpieces – already clamped on palettes – between operations in central or local buffers. The workpiece setup system is in charge of managing the clamping and unclamping of workpieces onto and off the palettes. Finally, the handling system performs necessary changes in the orientation of workpieces and palettes between the machines, the transportation and the setup systems.

Similar to the workpiece supply system, the tool supply system is decomposed into a transport system, a storage system, a setup system and a handling system. The supply of tools to the local tool magazines of the individual machines occurs either manually or automatically. In case of manual exchange of tools, the processing at the concerned machine must be interrupted which results in a relatively long time lost in setup.

The information system carries out all the automated functionalities required for running the technical system namely the short-term planning, scheduling and control. It usually entails a data management system for keeping record of and providing access to all data required for these functionalities.

The human operator system includes all the personnel directly required for the operation of the whole system. As illustrated in Figure 2.3, operators are directly controlling not only the information system, but also the technical system. This control involves tasks like workpiece preparation, tool setup, maintenance and repair.

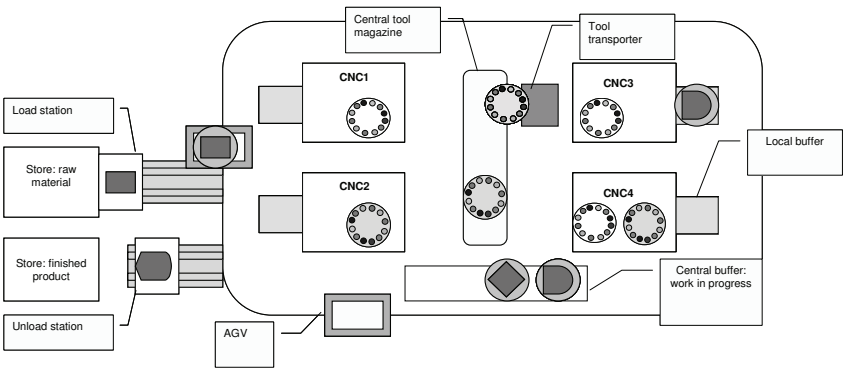


Figure 2.4: The layout of a typical FMS

Figure 2.4 illustrates the layout of a typical FMS. The processing system consists of four CNC universal machines. Each of these machines encompasses a local tool magazine equipped with different tools. The current setup of the machines with tools determines the operations that the machine can support as well as the part types that it can process. The workpiece supply system consists of a central buffer, two AGVs and load/ unload stations connected to inventory stores. The tool supply system consists of a central tool magazine and a special AGV for transporting tools.

2.2 Scheduling for FMS

Decisions and tasks concerning the management of production are categorized into three interrelated functions: planning, scheduling and control [Stec85]. These are captured in Figure 2.5, from a hierarchical view with the corresponding levels of the automation pyramid. Planning is basically concerned with decisions related to specifying the part types and quantities to be produced as well as allocating machines with tools to form the desired machine groupings. The request for producing a certain part type with a specified quantity represents a planned job. A plan delivered from planning to scheduling includes the set of jobs to be produced over a period of time. A schedule for executing this plan is generated by assigning jobs to resources in real time and is delivered to shop floor control for execution.

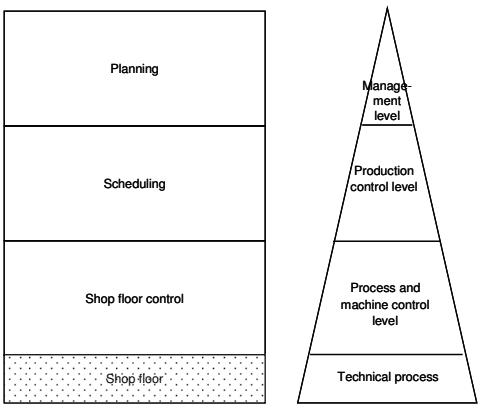


Figure 2.5: A hierarchical view of manufacturing control

2.2.1 Scheduling Environment

As depicted in Figure 2.5, scheduling is integrated in the manufacturing control functionalities within planning and shop floor control. A thorough understanding of scheduling cannot be attained without comprehending these two functionalities, forming together the environment of scheduling.

2.2.1.1 Planning

Planning is concerned with decisions that have to be made before the FMS starts production [Stec85]. The objective is to generate a plan yielding the product mix that would maximize the utilization of the available resources. To ensure the realization of planned jobs, the allocation of operations to machines have to be planned as well. This involves decisions on grouping

machines by allocating the available cutting tools to them. Machines are said to be pooled if they are identically tooled. Usually pooled machines are grouped together and support identical set of operations corresponding to their current setup with cutting tools [Stec85].

The generation of plans in FMSs is derived by customer requests in what is referred to as open shop environments. On the other hand, requirements are generated in closed shop environments based on supply, assuming that demand is deterministic [Grav81]. It follows that the nature of planned jobs in FMSs is stochastic and dynamic. In other words, variations in part types and volumes are very likely. In addition, planned jobs are subject to change, such as the cancellation or delay of a planned job or the modification in its quantity [VHL03]. In addition, since every job is associated with a single part type, different jobs can be planned for the same customer. This implies the existence of common constraints, such as the same delivery date, among different jobs.

2.2.1.2 Shop Floor Control

Shop floor control involves tasks performed after the release of jobs and during production to ensure that the schedule is being met as planned. This includes the continuous monitoring and data collection of the current status of the shop floor. In case of breakdowns, hindering the fulfillment of the planned schedule, actions – that minimize the deviation from the planned schedule – are to be taken by the control. In addition, preventive maintenance is usually performed during off-shifts. Inspection activities necessary for quality assurance are also part of the control process. Several problems detected by the control necessitate a revision in the schedule. This includes machine or tool breakdown, operator absenteeism, unavailability of material and rework [VHL03].

2.2.2 Definition of FMS Scheduling

Scheduling is concerned with the real-time planning of the production flow of the planned jobs under consideration of the current setup of the machines, determining the operations currently supported by each machine. The production flow of a typical workpiece consists of a set of production steps, as illustrated in Figure 2.6. After loading a workpiece, it has to be transported and stored in the input buffer of the machine, selected for performing the first operation for this workpiece. This is followed by processing the workpiece at this machine and storing it in the output buffer. Based on the work plan of the workpiece, it is either transported to the input buffer of another machine, for performing the next operation, or to the unload station[2].

As illustrated in Figure 2.6, these steps can be classified into processing steps and material handling steps. In this context, material handling corresponds to the provision of workpieces to

2 For simplicity, the temporary storage in work-in-progress buffers, which takes place between two transportation steps is disregarded in this illustration.

the required machines at the specified time. Accordingly, scheduling can be decomposed into two sub-problems: production scheduling, which is concerned with the processing steps, and transportation scheduling, which addresses the material handling steps [ELR06].

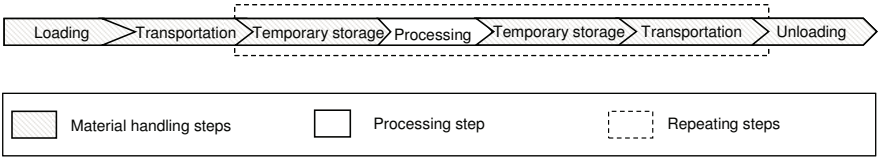


Figure 2.6: The production flow of a typical workpiece

In the rest of this Chapter, a definition of the FMS scheduling problem is given by defining the production scheduling and the transportation scheduling sub-problems.

2.2.2.1 Production Scheduling

Production scheduling is concerned with specifying for every operation of every job a set of machines to be executed at, over time. It can be further classified into the following sub-problems [WaLi02] [BaGö10].

Allocation

The allocation sub-problem deals with deciding, for every job, on a set of machines for the parallel execution of a given operation of this job on the selected machines. Machines currently performing the same operation may differ in their current load and thus can be associated with different start times. A decision on a certain allocation implies certain lot size planning of a given job quantity on the selected machines for the parallel execution. In solving this sub-problem, predefined performance criteria have to be considered such as minimizing the time taken by a certain job for the given operation.

Sequencing

This sub-problem is concerned with finding the best arrangement of the allocated jobs at a given machine. As previously mentioned, the supported part types span a wide spectrum and are thus grouped into part families based on their similarities in the physical dimensions and their predefined operations. Machines switching from a part type to another have to incur setup time which is classified into minor and major, depending on whether the consecutive parts belong to different types of the same family or to two different families respectively. While sequencing jobs on a certain machine, the associated setup time should be minimized.

2.2.2.2 Transportation Scheduling

The transportation scheduling problem is similarly decomposed into two sub-problems: AGV allocation and routing [QHHW02].

AGV Allocation

This sub-problem concerns the allocation of AGVs to incoming job requests. The workpieces of every job have to be picked up from a given source, which could be a load station, a temporary store, or the output buffer of a machine, to a given destination, which could be an unload station, a temporary store or the input buffer of another machine. AGVs have to be allocated to jobs under consideration of constraints like job priorities and deadlines. The goal is typically twofold: to fulfill job constraints and to improve the utilization of the system resources. Improving the utilization of system resources includes objectives such as minimizing the number of involved AGVs and minimizing the total travel time.

Routing

The goal of routing is to find the shortest-distance or the shortest-time path for each one of the allocated AGVs from its source all the way to its destination. In case no direct path is available based on the physical layout of the system, the scheduled load has to be transferred from the AGV in concern to another AGV which brings the load to its destination. This shows the interrelationship with routing and allocation. In addition, the selected route should be feasible in that it has to be deadlock- and conflict-free.

2.2.3 FMS Scheduling as an Optimization Problem

Scheduling for flexible manufacturing systems is usually described as an extension to the conventional job shop scheduling problem (JSSP). The conventional JSSP has been extensively studied in the last decades because of its well-known complexity as one of the most difficult combinatorial optimization problems [Gupt02] [Grav81]. The problem is stated as follows. A set of jobs are to be executed on a set of machines according to a prescribed order. The objective is to minimize the makespan, i.e. the time required for all jobs to finish execution [HaHo97]. In the JSSP, the machines required for every job are predefined. Therefore, the problem can be regarded as the sequencing of jobs on each machine.

The FMS scheduling problem is even more complex than the standard job shop scheduling problem. The added complexity stems basically from the flexibility of machines in performing operations. Each job is to be allocated to a predefined set of operations rather than machines. Since every operation can typically be performed by more than one machine at any point in time, the allocation of machines to jobs for a certain operation has to be solved as well. In what follows, the problem is investigated by focusing on the optimization objective of the constituent sub-problems and providing an insight into the associated computational complexity.

2.2.3.1 Optimization of Production Schedules

The production scheduling problem is a problem of jobs competing on limited number of machines. The performance of production scheduling is usually evaluated based on time-based measures. Examples of the most frequently applied measures include [VHL03]:

- Makespan: An important measure of throughput calculated by simply taking the difference between the start time of the earliest scheduled job and the completion time of the latest scheduled job. It aids in estimating the productivity per unit time by multiplying its reciprocal by the number of the considered jobs.

- Tardiness: A measure of compliance to the predetermined due dates measured by the difference between the actual completion time and the due date for each job in which the due date is not met. Usually the objective function is to minimize the average or the maximum tardiness of the set of jobs under considerations.

- Machine utilization: A measure which is based on calculating parameters like machine utilization rate amounting to the ratio of the processing time of the machine to the total availability time of the same machine. Similarly, the blocking time can be considered by estimating the portion of time, in which a certain machine was blocked, to the total availability time of this machine.

The classical optimization objective of FMS production scheduling corresponds to the minimization of the makespan, which implies the maximization of the resource utilization. The tardiness measure is rarely used in assessing the FMS production.

In what follows, FMS production scheduling is discussed from the optimization perspective by investigating the decisions to be taken by the sequencing and the allocation sub-problems and their impact on the quality of the resulting schedule. In addition, the computational complexity of the problem is estimated and illustrated. This discussion is based on the example, depicted in Figure 2.7. In this example, the machines M1, M2 and M3 are initially configured for part families 2, 2 and 1 respectively. These machines are to be allocated in time to three jobs: 1, 2 and 3, belonging to two different part families, under consideration of the current setup of the machines. The resulting schedule should minimize the makespan of the three jobs.

Optimizing the Sequencing Sub-Problem

Since the switching to a different job on a given machine is governed by a predefined setup time, every sequence yields a different temporal behavior. To illustrate, consider the example depicted in Figure 2.7. The sequence at M1 in schedule (a) yields two units of setup time, required to first switch the machine from its configuration to part family1, to which the first job in the sequence belongs, and then to switch the machine back to part family2 to execute job2

followed by job3. Evidently, for a solution of the sequencing sub-problem to yield an optimal makespan, the setup time associated with the resulting sequence has to be minimized.

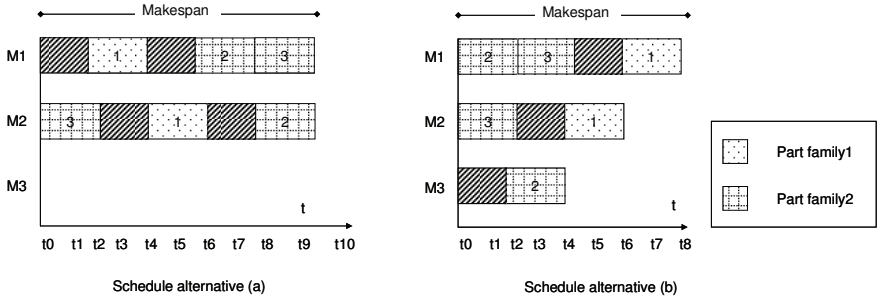


Figure 2.7: Examples of production schedule alternatives

The sequencing sub-problem is thus an optimization problem, whose solution space involves all possible permutations of the jobs to be scheduled on the concerned machine. For example, if the set of jobs to be scheduled on the machine is {1,2,3}, the solution space includes 6 possible sequences, such as (1,2,3) and (1,3,2). In general, the solution space amounts to n!, where n is the number of jobs to be scheduled on the concerned machine. Obviously, the associated complexity explodes by increasing the number of jobs. To illustrate, while the solution space for sequencing 10 jobs on a single machine amounts to around 3.6×10^{6}, the solution space for 15 jobs is approximately 1.3×10^{12}. The problem gets even more complex when considering the sequencing of the entire set of the machines.

Optimizing the Allocation Sub-Problem

Due to the stochastic nature of the arrival of jobs to be scheduled, the allocation of machines to jobs has to be optimized in a way that takes possible fluctuations in the part types of the incoming jobs into consideration. In other words, allocating the available resources to the incoming jobs should account for the uncertainty about the part types and quantities of further jobs that should be incorporated into the existing schedule. To illustrate, consider the two schedule alternatives depicted in Figure 2.7. While schedule (a) results in a longer makespan than schedule (b), the former preserves the initial ratio of the machine configuration, i.e. two machines for part family2 and one for part family1. Schedule (b), on the other hand, results in the changeover of the three machines in a way that changes the relative configuration to two machines supporting part family1, namely M1 and M2, and one machine supporting part family2, which is machine M3. In case the arrival probability of jobs from part family2 is expected to be relatively higher than that of part family1, schedule (b) may yield a longer makespan on the long run. Therefore, the effect of the allocation on the machines has to be taken into account to guarantee the minimization of the makespan.

The solution space includes all possible combinations of a given number of machines out of the total number of available machines. Each job can be allocated to the entire set of the available machines or to a subset of these machines. Consequently, the number of possible allocations of a total number of n machines to a single job, for performing a given operation, is calculated as in (2.1). Evidently, the complexity of the problem gets even higher when considering the allocations of all the available jobs and all the corresponding operations.

$$\alpha(n) = \sum_{i=1}^{n} \binom{n}{i} = \sum_{i=1}^{n} \frac{n!}{i! \cdot (n-i)!} \tag{2.1}$$

It is worth mentioning that the decomposition of the production scheduling problem into two sub-problems is just for illustration purposes. The problem is conventionally solved by simultaneously optimizing the sequencing and the allocation sub-problems. This results in an NP-complete problem, which means that there is no known algorithm that can solve this problem in a polynomially bounded computational effort [Gupt02].

2.2.3.2 Optimization of Transportation Schedules

In generating transportation schedules, the main objective is to minimize the total transportation time required to fulfill the requested jobs. This is attained by optimizing both the allocation and the routing sub-problems.

Optimizing the Allocation Sub-Problem

The allocation sub-problem is solved by assigning the requested jobs to the available AGVs. It is similar to the allocation of machines to jobs and is thus associated with a solution space, whose size is calculated by $\alpha(n)$ in (2.1), where n represents the number of available AGVs.

Optimizing the Routing Sub-Problem

The routing sub-problem can be regarded as a pathfinding problem, aiming to find a path from the source to the destination with the minimum associated cost. In this context, the cost is measured in terms of the needed transportation time, which is derived from the speed of the AGV and the length of the path. The complexity of this sub-problem depends on the physical layout of the shop floor and the degree of connectivity among the machines. The more the available paths from one machine to the other, the higher the complexity of the problem would be.

2.2.4 FMS Scheduling as an Iterative Dynamic Process

The previous discussion of the FMS scheduling problem has dealt with it from a theoretical perspective by focussing on defining the constituent sub-problems and explaining the way they are optimized. In practice, FMS scheduling is dealt with as an iterative dynamic process.

Therefore, the objective is not only to optimize the generated schedule but also to adapt it to unforeseen dynamic events. The process of updating an existing schedule in response to disturbances or other changes is referred to as dynamic scheduling or rescheduling [VHL03]. The objective of rescheduling is to optimize the modified schedule, as explained in the previous section and at the same time to minimize the deviation of the modified schedule from the original schedule [OCP03] [RFK04]. Minimizing the deviation from the old schedule is intended to avoid causing additional disturbances to other planned activities. This added requirement on dynamic scheduling is referred to as stability.

This section addresses the FMS scheduling problem from a practical perspective by giving an insight into the dynamics of the scheduling environment and explaining strategies and methods used to deal with these dynamics.

2.2.4.1 Dynamics of the Scheduling Environment

The scheduling environment, represented in planning and shop floor control, features nondeterministic dynamic events that cause rescheduling. These events can be classified into disturbing and non-disturbing events. Non-disturbing events correspond to the arrival of jobs that have to be incorporated into the schedule without causing any change to the already scheduled jobs. In other words, they cause updating the schedule but do not cause modifications to the already scheduled jobs. This is the case if the priorities of the newly coming jobs are lower than the priorities of the existing jobs. On the other hand, disturbing events, also called disturbances, are those events that are associated with inevitable or potential modifications to the already scheduled jobs. These events are triggered either during planning or during shop floor control. Examples of disturbances that stem from planning include the incoming of an urgent job and the cancellation or the delay of an already scheduled job. Disturbances occurring during shop floor control include the failure of a resource and the over- or underestimation of the processing time.

2.2.4.2 Rescheduling Strategies

There are two main strategies for tackling the dynamics and uncertainties of the FMS scheduling environment [VHL03].

Completely Reactive Scheduling

According to this strategy, scheduling decisions are taken at the control level by applying decentralized methods for dispatching jobs, waiting on available resources. These control methods employ dispatching rules or other heuristics for dynamically prioritizing competing jobs. When a machine becomes ready, it selects from its local queue the job having the highest priority to be next processed. In other words, no schedule is generated a priori, and jobs are assigned to machines dynamically based on the current status of the system.

Predictive-Reactive Scheduling

This is the most widely applied strategy in dealing with dynamic events. It works by first generating a schedule and then updating it whenever necessary. In updating the schedule, minimizing the impact of change is aimed at, which is realized by following different policies and methods of rescheduling. Policies are concerned with the point in time in which rescheduling takes place. Three different policies are proposed in literature: periodic, event-driven, and hybrid. According to the periodic policies, the schedule is updated on a periodic basis, which is referred to as the rolling time horizon basis. This periodic revision of the schedule corresponds to decomposing the dynamic scheduling problem into a series of static scheduling problems that can be solved by the conventional scheduling techniques. The event-driven rescheduling policy revises the schedule upon the occurrence of disturbances altering the system status. Based on the hybrid policy, a schedule is revised periodically and upon the occurrence of major events like machine breakdown and the arrival of urgent jobs. In other words, while regular events are delayed to the next rescheduling period, major events trigger a revision of the schedule [VHL03] [OuPe08]. However, the basis for classifying events into urgent and non-urgent is not made clear.

2.2.4.3 Rescheduling Methods

The predictive-reactive strategy applies one of three methods in reacting to dynamic events. In what follows, these methods are described.

Generating Robust Schedules

This method is concerned with generating a schedule which accounts for unforeseen events. A schedule is said to be robust if it optimizes efficiency measures and at the same time yields low degrees of deviation in case of disturbances. The concept seems interesting since it leads to generating a schedule that can absorb unexpected events with minimal adjustments while maintaining good system performance [VHL03]. However, the way this robust schedule can be generated is not easily identifiable in literature [OuPe08].

A common approach for generating robust schedules is the insertion of idle time slots in the schedule, typically between subsequent jobs, to account for potential contingencies [VHL03]. Apparently, this approach can reduce the number of schedule revisions. Nevertheless, it can lead to degrading the resource utilization by keeping the machine idle during the inserted buffers. Moreover, planning for the unforeseeable events can be almost impossible in manufacturing systems with highly stochastic and dynamic environments like FMSs. This is confirmed by a simulation study which shows that the generation of a robust schedule outperforms conventional priority rules with respect to a set of situations but in case of increased processing time variability the performance of robust scheduling is shown to decrease remarkably [VHL03].

Repairing Schedules

This method involves applying adjustments to the planned schedule in reaction to dynamic events in a way that retains the old schedule. The repair is carried out through manual interventions by operators. This is performed based on repair methods like right-shift rescheduling and partial rescheduling. The two methods are based on applying delay to the planned schedule in reaction to events like machine breakdowns. While the former method applies changes to all the remaining operations by the amount of the downtime, partial rescheduling delays only the operations affected directly or indirectly by the disturbance [VHL03] [OuPe08].

Repairing the schedule has the advantage of retaining a good stability. However, the overall efficiency can be degraded, due to the tendency to repair the schedule at the local level concerned by the disturbance.

Regenerating New Schedules

According to this method, the old schedule is completely disregarded and a new schedule is regenerated in reaction to a disturbance. In regenerating a new schedule, all jobs that are not processed before the rescheduling point are included in the new schedule regardless of whether they are affected by the disturbance or not.

Regenerating a completely new schedule in reaction to disturbances could maintain high efficiency rates due to the comprehensive consideration of the entire system. However, this method is rarely applied in practice due to the extensive computational complexity which degrades the reactivity to the environmental dynamics. Moreover, generating a schedule from scratch each time a disturbance occurs would lead to a lack of continuity in processing the planned schedule and to poor stability, the so called shop floor nervousness [OuPe08].

2.3 Requirements on FMS Scheduling

FMS scheduling has to account for the special characteristics of the system as well as the dynamic nature of the environment. Consequently, a scheduling solution has to satisfy the following requirements.

2.3.1.1 Efficiency

FMS scheduling is an optimization problem aiming to make the best use of resources to achieve the predefined objectives. Optimization is based on multi-criteria, defined during the planning phase. These criteria usually correspond to increasing the productivity of the system. However, the focus can be laid on different aspects depending on the planning tactics. Example objectives include minimizing the makespan and balancing the load on the machines. Towards increasing

the productivity of the FMS, the exploitation of the available component flexibilities should be maximized. In addition, the efficiency of FMS schedules should be assessed by how far they satisfy the involved customers. In this context, customer satisfaction is judged in terms of the fulfillment of the timeliness requirements imposed on the jobs belonging to a given customer.

2.3.1.2 Short-Term Flexibility

FMS scheduling is required to show short-term flexibility in reaction to the dynamics of the scheduling environment. By reflecting on the manufacturing flexibility taxonomy, presented in section 2.1.1, it becomes evident that process, volume and routing flexibilities fall under this category. Consequently, satisfying short-term flexibility necessitates accounting for the stochastic nature of planned jobs by handling variations in part types and quantities as well as variations in the number of jobs delivered to scheduling at any point in time. In addition, scheduling has to account for unforeseen events and show robustness to contingencies by exploiting the redundant capabilities of the resources. This is not possible without exploiting the available components flexibility by efficiently managing the collective capabilities and resources. For example, handling a failure in one of the resources is only possible if other resources, possessing the same capabilities, are taken into account. Moreover, scheduling has to be reactive by ensuring the desired response within real-time limits. This reactivity is essential to enable realizing FMS scheduling as an ongoing process interleaved with planning and control.

2.3.1.3 Long-Term Flexibility

Changes to the structure of the underlying shop floor or to the planned products – by for instance introducing a new product or adding a new machine – is required to be incorporated easily without complex or major modifications. This category compromises product and expansion flexibilities, addressed in section 2.1.1. Extending the existing FMS by introducing modifications to the planned products as well as through structural changes to the shop floor should be accommodated by scheduling dynamically. In addition, dynamic changes concerning scheduling objectives and methods should be made possible.

2.3.1.4 Integration Ability

Due to the tight connection of scheduling with its environment represented in planning and control functionalities, FMS scheduling has to support the integration within these relevant functionalities. The integration with shop floor control should provide support for the automated capturing of the entire set of resources involved in scheduling. Data exchanged between scheduling and its environment should be propagated in real-time. The integration with planning should be made in a way that gives the decision makers an updated view of the current state of scheduling and allow them to intervene when necessary.

In this chapter, the necessary knowledge for conceiving FMS scheduling has been presented. The first part of the chapter was devoted to developing understanding of flexible manufacturing systems by emphasizing the concept of manufacturing flexibility as an influential factor in recognizing the significance and peculiarities of FMS scheduling. Based on this discussion, FMS scheduling problem has been discussed by defining it and discussing the way it is handled from the theoretical and the practical perspectives, by stressing the efficiency of the generated schedules and the flexibility of the scheduling process respectively. The final part in this chapter derived the requirements on FMS scheduling in light of the knowledge developed about both FMSs and their scheduling problem.

3 Survey of FMS Scheduling Approaches

The FMS scheduling problem has been extensively studied throughout the decades resulting in promising theoretical results and well-established approaches. However, this theoretical endeavor has found limited applicability in practice. This chapter is devoted to investigating and assessing the theoretical approaches from a practical viewpoint. Due to the discrepancy between theory and practice, this survey has not been a straightforward task. To deal with this problem, the survey adopts a three-fold evaluation framework covering the scheduling perspective, the scheduling architecture and the scheduling technique. The assessment of the different approaches is based on the requirements, derived in chapter 2.

This chapter is organized as follows. Section 3.1 embarks on the conventional scheduling approaches with respect to their perspective in capturing real FMS shop floors. Section 3.2 examines the various architectures with the goal of finding out the architecture that matches with the characteristics of FMSs. For this purpose, a taxonomy of architectures is proposed and used to assess the surveyed architectures. Section 3.3 examines the scheduling literature from the perspective of the applied techniques. Section 3.4 points out concluding remarks derived from the assessment of the different approaches.

3.1 Surveying Approaches based on Scheduling Perspective

Different scheduling studies have addressed the problem from different perspectives. The adopted perspective relates to two main dimensions: the scheduling scope, or the extent of coverage to the shop floor resources, and the integration mode in the environment. The scheduling scope can be either machine-centric, in which production schedules are generated and transportation schedules are disregarded, or comprehensive, where production and transportation schedules are integrated. The integration mode in the environment can take one of two values: either static or dynamic. While the former is based on the assumption that the scheduling problem is isolated from its environment, the latter takes the environmental changes into account while generating schedules. Accordingly, conventional scheduling approaches can be classified based on these two dimensions into four different categories, derived in Table 3.1. In what follows, these classifications are explained.

3.1.1 Static Production Scheduling Approach

Research under this category has been devoted to machines or resources with processing capabilities, totally ignoring other relevant production resources. Researchers have considered the problem from a theoretical point of view by isolating it from its environment and making

hypothetical assumptions. Despite its limited scope and thus its limited practical influence, research under this category has contributed in laying the foundation of the scheduling theory. In addition, the attained results have acted as benchmarks with respect to the computational complexity of the considered algorithms. However, results of this research category have hardly witnessed application in practice. Examples of scheduling research in this category are found in [WaLi02], [PRMP07] and [SMCS09].

Table 3.1: Classifying scheduling approaches based on the scheduling perspective

	Scope		
	Machine-centric		**Comprehensive**
Integration mode	**Static**	Static production scheduling approach	Static integrated scheduling approach
	Dynamic	Dynamic production scheduling approach	Dynamic integrated scheduling approach

3.1.2 Dynamic Production Scheduling Approach

In an attempt to narrow the gap between theory and practice, dynamic scheduling or rescheduling came into existence. Examples of dynamic production scheduling are found in [AbSv97], [KPM97], [SuXu01], [CoJo02], [SaKi03] and [Ouel03], [RFK04], [DuPe07]. Online rescheduling is either centralized [SuXu01] [CoJo02] [SaKi03] or decentralized [KPM97] [Ouel03]. While centralized solutions suffer from a low reactivity due to their excessive computational complexity, the proposed decentralized solutions are mostly specific solutions which hinders their applicability in practice. This limited applicability of the dynamic production scheduling research is manifested by the prevalence of informal ad hoc scheduling repair methods in practice [Herm06].

3.1.3 Static Integrated Scheduling Approach

A number of attempts are reported in literature, for integrating transportation scheduling with production scheduling. Research work under this category isolates the problem from its environment by investigating the scheduling of a predetermined set of jobs under consideration of a defined set of parameters. In [JAPS98], [ELR06], [JASD06] and [THM09], different algorithms for integrating transportation scheduling with production scheduling are proposed. These algorithms are tested on simple FMS configurations with no more than 6 machines and two AGVs. These algorithms aim at minimizing the makespan.

A thorough analysis of the surveyed approaches coupled with the results reported in a recent state of the art survey [ElKh07] has revealed the following observations [BSG10]:

- The requirement on enhancing efficiency is always addressed at the cost of short-term and long-term flexibility which are rarely addressed.
- The considered FMS configurations are based on simplified assumptions which don't hold in real configurations. Adopting the proposed concepts for a different configuration is usually a tedious task due to the poor abstraction from the assumed system configuration.

3.1.4 Dynamic Integrated Scheduling Approach

Dynamic integrated scheduling systems have been always targeted in flexible manufacturing systems. Traditionally, this has been attempted via the computer integrated manufacturing (CIM) approach. This approach is characterized by its comprehensive scope to interrelated manufacturing tasks like design, planning and scheduling. The agent-based approach has been advocated as a suitable approach for overcoming the drawbacks of the traditional CIM approach and tackling the inherent complexity of the manufacturing environment through the flexibility exhibited by software agents. In what follows, these two approaches are briefly reviewed.

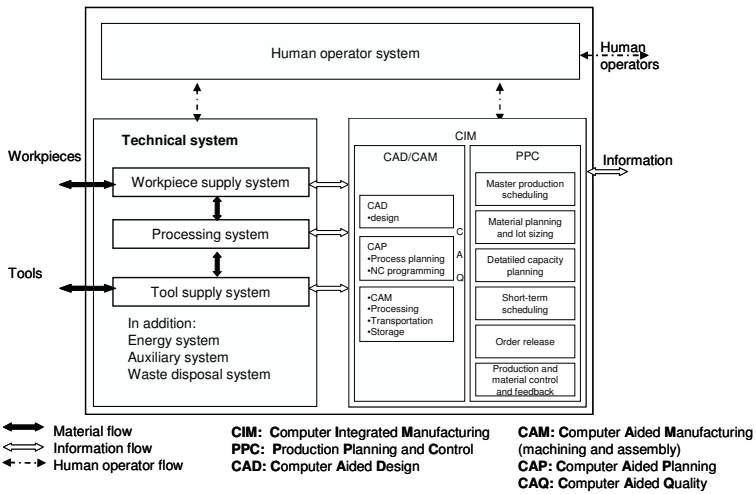


Figure 3.1: CIM applied in a typical FMS – adapted from [TeKu93]

3.1.4.1 Computer Integrated Manufacturing Approach

The highly integrated nature of the planning and control decisions of FMSs has motivated researchers and practitioners to adopt the CIM concept. CIM consists of software tools for computer aided design (CAD) as well as for production planning and control (PPC). Traditionally, the concept of CIM has been widely applied in FMSs by realizing it on a

centralized computer connected to the technical system, as depicted in Figure 3.1. However, the conventional CIM realizations have failed to achieve the desired integration with the technical system and their role has been limited to providing support to the user as other decision-support systems [MoYa07]. With the advancement of technology and the changes in the market, a higher demand has been put on a more sophisticated integration that enables a higher degree of automation with enhanced responsiveness to the market changes.

3.1.4.2 Agent-Based Approach

Another research direction for achieving the desired dynamic integrated scheduling is represented in the application of the agent-based approach, which possesses very good potential for tackling the inherent complexity of the FMS scheduling problem and exhibiting the required flexibility[3] [SWH06] [GuZh10]. In [BuSi01], an agent-based system for an automotive engine assembly line belonging to Daimler AG is presented. This system was developed to support the introduction of flexible buffers to a conventional linear assembly line along with multi-function stations. Physical resources in this assembly line were represented by agents, such as docking stations, AGVs and engine buffers. These agents work on finding the best route for every incoming engine by autonomously taking scheduling decisions at the control level. Test results have shown remarkable enhancement with respect to robustness and scalability.

In [JLMF07], Saarstahl AG, a German steel manufacturing company, reports on its experience with developing and deploying an agent-based planning and scheduling system and shows how the agent-based system could enhance the throughput within uncertainties and change. The main task of the system is to adapt a given daily schedule to the disturbances that disrupt the planned execution under consideration of system-specific constraints. The system is structured in a three-tier architecture corresponding to a database capturing an updated snapshot of the steelwork, a planner carrying the responsibility of the overall correction of the schedule and the agents representing the system resources.

In spite of the strong belief in the gained benefits of agents in FMS manufacturing scheduling [JoRa98] [SWH06] [Leit08], the proposed agent-based solutions are specific solutions targeted for a certain FMS. This is attributed to the lack of agent-based scheduling solutions, addressing FMSs from the dynamic integrated perspective in a way that can be applied for any FMS. The conception of a generic agent-based solution for FMS scheduling necessitates incorporating agents with suitable scheduling techniques and organizing them in suitable architectures [SWH06].

[3] A detailed discussion of the concept of agency is given in the next chapter.

3.1.5 Assessment of the Surveyed Approaches

It can be concluded that static approaches are not suitable for FMSs, whose environment is highly dynamic. Adopting a static approach in practice would necessitate increased manual intervention to adjust the schedule to the current environmental status. In addition, this would lead to decreased utilization of the available resources and a decreased quality of the generated schedule. A dynamic approach is thus required to suit the dynamics and automated control of FMSs.

The routing flexibility of FMSs calls for an integrated approach that is capable of integrating transportation scheduling with production scheduling. It has been shown that the CIM approach can only aid in realizing decision-support systems and have failed to provide the required integration with the environment. On the other hand, it has been shown that the agent-based approach possesses a high potential for dealing with the problem complexity flexibly. However, the adoption of the agent-based approach in FMSs calls for suitable architectures in addition to scheduling techniques supplementing the selected architectures.

3.2 Surveying Approaches based on Scheduling Architecture

An architecture specifies how data and control are organized and how the individual software components interact with each other. Therefore, it plays a significant role in specifying the overall system behavior especially with respect to the reactivity and extensibility. Developing a scheduling architecture has to account for the requirements on efficiency, flexibility and integration ability. In this section, the scheduling approaches are surveyed and evaluated from the point of view of the adopted architecture. This survey is based on a taxonomy which presents a classification of architectures.

3.2.1 Taxonomy of Architectures

In this context, an architecture addresses not only the organization of software components but also the way the scheduling problem is decomposed into components. In other words, there are two factors affecting the development of a system architecture: the problem decomposition and the organizational structure. In what follows, these two factors as well as the resulting classifications are explained in details.

3.2.1.1 Problem Decomposition

The complexity of a real problem is tackled in software solutions by decomposing the problem into smaller more manageable entities. In the manufacturing domain, two decomposition arts are possible: functional decomposition and physical decomposition [Shen02b]. Under functional

decomposition, the problem is represented in terms of software entities corresponding to functions such as monitoring, allocation, sequencing, etc. On the other hand, physical decomposition maps the objects in the physical world into software entities. Examples of physical entities include machines, tools, jobs and operations.

3.2.1.2 Organizational Structure

The task of selecting an organizational structure is concerned with defining the interrelationships and interactions among the decomposed software entities. Manufacturing scheduling architectures can be classified according to this dimension into four categories [DBW91] [Shen02b], which are schematically depicted in Figure 3.2.

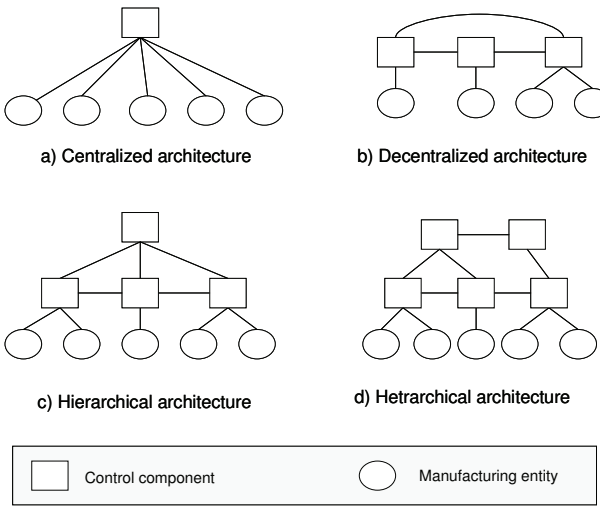


Figure 3.2: Organizational structures of scheduling architectures – adapted from [DBW91]

The centralized organizational structure assigns all decisions concerning scheduling to a single control component connected with the concerned manufacturing entities at the shop floor. This component usually bases its decisions on updated information, capturing an updated view of the shop floor control and collected from the concerned resources [DBW91].

Decentralized architectures decompose the problem into components, that are delegated the scheduling and control of a set of physical resources. These components are allowed to communicate but they have the same privileges and no one has the right to control the other [Shen02b].

According to the hierarchical architectures, the problem is decomposed into components with different hierarchical privileges. A supreme component at the highest level of the hierarchy is

entitled the global supervision and coordination of tasks. While the delegation of tasks flows downwards, reports about the current status of operation flows upwards. The communication among the components at the different levels follows the master-slave paradigm. The poor reactivity and communication rigidity associated with the master-slave paradigm led to allowing components at a certain level to communicate together, resulting in the structure depicted in Figure 3.2 [DBW91].

Heterarchical architectures are a modified form of the hierarchical architectures in which components are more autonomous in selecting their actions at their local level. The communication among entities is also more flexible and takes the form of negotiation to fulfill a common task rather than the form of orders, from the supreme components to the subordinate components. Accordingly, no supreme component is necessary [DBW91].

3.2.2 Centralized Approach

Most of the static solutions, surveyed in section 3.1, adopt a centralized architecture. Examples for this research can be found in [Witt90], [AbSv97], [RoDi00] and [WaLi02]. Centralized architectures are mainly associated with the advantage of concentrating the control in one entity, which possesses complete information about the status of the FMS. This concentration facilitates optimization and ensures consistency of the generated schedules. However, it suffers from poor reactivity and extensibility due to the monolithic structure.

3.2.3 Decentralized Approach

This approach aims mainly at fulfilling the required reactivity. The decentralized schedule generation usually depends on the application of simple priority rules at the control level. Schedules are normally generated at the local level of each machine, which reduces the efficiency at the global level. The predominant decomposition art is the physical decomposition due to its natural one-to-one mapping to the shop floor entities. A physically decomposed decentralized architecture is adopted in [BuSc01], where software agents represent entities in the shop floor such as workpieces, switches and machines. The system is mainly intended for the shop floor control. However, simple scheduling functionalities like routing are also supported. This is enabled through the interaction among workpiece agents and other concerned agents including switch agents and machine agents. In [Aydi07], a functionally decomposed decentralized architecture is described. According to this architecture, scheduling is decomposed into several agents, each of which undertakes the optimization of the generated schedule by applying a different scheduling technique. These agents coordinate their optimization of the schedule based on a shared data repository.

3.2.4 Hierarchical Approach

This approach attempts to combine the efficiency of the centralized approach with the reactivity of the decentralized approach by optimizing schedules at different levels of abstraction. In [TeKu93], a hierarchical architecture for planning, scheduling and control of FMSs is described. It consists of components representing the control functionalities at different levels of abstraction for the different physical cells of an FMS. Similarly, every manufacturing cell in the shop floor is represented by a component in the hierarchical architecture described in [WXZ08]. Every component is responsible for all scheduling tasks associated with the corresponding cell. In [KPM97], a physically decomposed hierarchical architecture is proposed. This architecture emphasizes reactivity by organizing components at only two levels of abstraction. Components at the lower level are responsible for the dynamic scheduling of a given work centre. A supervisory component, at the upper level, monitors the global state of the shop floor and the current status of the subordinate components. It intervenes by ordering subordinate components to apply a certain modification to their schedules, whenever a modification is necessary to enhancing the overall performance.

3.2.5 Heterarchical Approach

According to this approach, components are usually represented as agents to exhibit the required autonomy and interaction flexibility. In [Shen02a], a scheduling system is proposed in which shop floor resources are represented at three different abstraction levels. At the lowest level, machines and workers are represented as agents. At the higher level, an agent for coordinating all machine agents and another one for coordinating worker agents are introduced. These two coordinator agents are again coordinated by another agent located at the highest level of abstraction. The communication between agents at the different levels takes the form of negotiation and cooperation. In [VaMa00], another heterarchical architecture is described. In this architecture, agents involved in FMS scheduling are classified into management agents and machine agents. While machine agents are in charge of optimizing the schedules of their machines, management agents act as mediators between machine agents and customer requests.

3.2.6 Assessment of the Surveyed Approaches

In this section, the surveyed architectures are evaluated in light of the aforementioned taxonomy and based on the requirements, derived in the previous chapter. The results of the assessment are shown in Table 3.2, where the used notation indicates whether the criterion is completely satisfied (++), satisfied (+), partially satisfied (0), or not adequately satisfied (-). Only centralized architectures are not analyzed based on the decomposition art since it does not apply to them.

As illustrated in Table 3.2, the centralized architecture is distinguished by its good potential to generate an efficient solution. Attempting to carry out the scheduling functionality for the entire shop floor by only one software entity is overwhelming and results in poor reactivity to changes in addition to poor fault tolerance due to the existence of a single point of failure. The shortcoming of the centralized architectures becomes even more evident when considering the long-term flexibility. In other words, modifiability as well as extensibility is only possible with relatively high effort. The integration ability with the manufacturing environment is poor because of the lack of modularity. Capturing every single event at the centralized level is hard especially in systems with highly dynamic environments like FMSs.

Table 3.2: Evaluation results of scheduling approaches based on their architecture

		Efficiency	**Short-term flexibility**	**Long-term flexibility**	**Integration ability**
Centralized approach		++	-	-	-
Decentralized approach	Physical decomposition	-	0	+	++
	Functional decomposition	0	0	0	+
Hierarchical approach	Physical decomposition	+	0	+	+
	Functional decomposition	+	0	0	0
Heterarchical approach	Physical decomposition	0	+	++	++
	Functional decomposition	0	+	+	+

The decentralized approach is characterized by a relatively better short-term flexibility, due to the enhancement of reactivity, in comparison to the centralized approach. However, the short-term flexibility is partially satisfied due to the difficulty to exploit the collective capabilities of resources based on the decentralized control. Any attempt to compensate this drawback by increasing the communication among the component would lead to higher complexity and thus to reduced reactivity. On the other hand, the decentralized decision making leads to reduced efficiency since the generated schedule is optimized out of the local perspectives of the components. The efficiency is relatively higher in case of functional decomposition due to the possibility of performing functionalities such as the allocation of resources in a more comprehensive sense. However, physically decomposed decentralized architectures outperform functionally decomposed decentralized architectures in terms of the long-term flexibility and the integration ability. Obviously, the long-term flexibility of the decentralized approach is relatively good because of its natural mapping of the real entities to software components, which facilitates incorporating modifications and extensions to the FMS. Similarly, the integration ability of the decentralized approach is satisfied since it is characterized by scheduling decisions that emphasize reactivity more than optimality and are based on simple parameter values that can be easily captured dynamically.

It is worth noting that, regardless of the organizational structure, the decomposition art affects the long-term flexibility as well as the integration ability. Physically decomposed architectures are easily manageable and extensible due to the one-to-one mapping between the software components and the real entities. In this way, the definition of a software component allows for encapsulating attributes of the physical entity, which makes modifications in the FMS much easier to incorporate in the software architecture. With regards to the integration ability, physically decomposed architectures are easily integrated in the manufacturing environment because of the natural correspondence between the physical and the software entities.

The performance of hierarchical approach lies in the middle range between the centralized and the decentralized approaches, with respect to efficiency. In terms of short-term flexibility, they are characterized by low reactivity but possess a better potential than the decentralized approach in exploiting the collective capabilities of the resources. Intuitively, the modularity of the hierarchical approach leads to acceptable performance with respect to the long-term flexibility. Similarly, the integration ability of the hierarchical approach is at least partially satisfied, which is attributed to the existence of distributed entities capable of the parallel detection of the simultaneous events on the shop floor.

As illustrated in Table 3.2, the heterarchical approach is characterized by low efficiency due to the myopic nature of the autonomous components. On the other hand, the heterarchical approach is characterized by enhanced flexibility. While the enhancement of the short-term flexibility is attributed to the higher autonomy and decision making freedom of their software components, the improvement in the long-term flexibility stems from the flexible interactions among the components, which facilitates incorporating new components dynamically. The integration ability of the heterarchical approach is better than that of the hierarchical approach due to the flexible interactions among the software components.

Based on this discussion, it can be concluded that none of the surveyed approaches fulfill all the requirements. Obviously, heterarchical architectures provide a flexible framework but they need to be extended in a way that enhances their efficiency and short-term flexibility. Satisfying the short-term flexibility to its utmost calls for incorporating a heterarchical architecture with methods for adapting the schedule to unforeseen events.

3.3 Surveying Approaches based on Scheduling Techniques

The manufacturing scheduling problem has been receiving increasing interest from researchers along the decades. This has resulted in a considerable body of research especially focused on the methodological aspect. The scheduling literature is very rich in the applied techniques and the benchmarking studies. Instead of embarking on the individual proposed techniques, a classification is presented and example techniques under every classification are highlighted and referenced.

3.3.1 Mathematical Approach

Mathematical techniques have been extensively applied to the job shop scheduling model, which is the theoretical model capturing the characteristics of FMSs. Applying mathematical techniques for analytical purposes has given researchers a more profound insight into the problem. This approach depends on formulating the problem with mathematical techniques like integer programming, mixed-integer programming and dynamic programming. This involves mathematically formulating all the decisions with their associated parameters, constraints, and optimization objectives. This is associated with high complexity which is tackled in theory by introducing simplifying assumptions, which normally do not hold in practice. A foundational study is conducted in [Stec83], which resulted in decomposing the scheduling problem into five interrelated problems. Despite the theoretical and analytical significance of this approach, it has had no remarkable impact in practice [BaMi94], [JoRa98].

3.3.2 Dispatching Rules Approach

Dispatching rules are rules of priorities assigned to jobs waiting on a common resource based on a certain criterion. Due to their simplicity and relatively low computational complexity, dispatching rules have been adopted on a large scale in practice. The number of existing rules is overwhelming and each of which is suitable for a certain environmental setting. The difficulty in selecting the suitable rule is attributed to the fact that there are $n!$ ways of sequencing n jobs waiting in the queue of a particular resource [RaHo99]. This raises the question of how to select the best rule dynamically. Moreover, most of these rules consider the local queue of the resource under consideration, which leads to decreased performance on the global scale. In [KPM97], an agent-based scheduling system, applying different dispatching rules for schedule generation and update, is proposed. Dispatching rules are dynamically selected based on the current status of the environment using a knowledge base. This knowledge base consists of simple if-then-else rules. The selection of the suitable dispatching rule is also made possible by employing simulation-based systems for evaluating the expected performance of candidate rules. In [Smit03], a survey of the use of simulation in FMSs is presented. Undoubtedly, assessing the performance ahead of selecting a certain rule results in enhanced efficiency but is associated with other drawbacks that will be clarified in the assessment section.

3.3.3 Artificial Intelligence Approach

3.3.3.1 Knowledge-Based Approach

The complexity of the scheduling problem and its dependency on human expertise acted as a motivation to this approach. Knowledge-based systems tackle the problem by capturing the human expertise and employing an inference engine that can deduce conclusions from the

existing knowledge to aid in reacting to dynamic events. A number of knowledge-based systems have been applied to the job shop scheduling problem like ISIS, MPECS, and OPIS. These systems usually apply a hierarchical constrained-directed search that takes organizational goals and physical limitations into consideration [JoRa98].

Simulation techniques are usually integrated with knowledge-based systems for selecting among a set of alternatives recommended by the knowledge-based system [JoRa98]. This integration can enhance the overall efficiency. Nevertheless, the complexity involved in the central reasoning about the whole shop floor and simulating decision alternatives hinders the application of this approach in dynamic manufacturing environments.

3.3.3.2 Search Heuristics Approach

This approach applies heuristics to tackle the computational complexity of the FMS scheduling problem. Heuristics are algorithms that might not always find the best solution but are guaranteed to find an acceptable solution in reasonable time. Heuristics that are widely applied to the FMS scheduling problem include algorithms like tabu-search, simulated annealing and genetic algorithms (GAs) [OuPe08]. All these algorithms apply heuristics in searching for a near optimal solution. While both tabu-search and simulated annealing are based on manipulating one feasible solution, GAs keep a population of solutions to iteratively generate offspring population. This feature has distinguished GAs with more resistance to premature convergence on local minima [LGP97]. Examples of applying GAs to FMS scheduling are found in [Wall96] and [MaBi04].

3.3.3.3 Other Approaches

These approaches apply other techniques from the AI literature. These include neural networks and fuzzy logic [JoRa98]. However, the work reported on these techniques is so limited and allows no further assessment. They are mentioned here for the sake of completeness.

3.3.4 Assessment of the Surveyed Approaches

The results of evaluating the surveyed techniques are listed in Table 3.3. As illustrated in the table, the mathematical techniques are associated with poor efficiency because they fail to capture reasonably sized problems and their complexity grows dramatically when it comes to real industrial examples [BaMi94]. Nevertheless, their efficiency is evaluated as neutral because this drop in performance is balanced by their good capability to account for the entire set of resources [Stec83]. Moreover, the short-term flexibility as well as the integration ability of the mathematical approach is not adequately satisfied. The integration of mathematical techniques in a real manufacturing environment is not straightforward since their formulations presuppose the existence of a large number of parameters, whose values are hard to capture dynamically.

However, the long-term flexibility is relatively better because mathematical techniques can be decomposed as reported in [JoRa98]. This decomposition is yet restricted with constraints and it is not clear how to develop flexible communication pattern among the decomposed entities.

Table 3.3: Evaluation results of the scheduling techniques

	Efficiency	**Short-Term Flexibility**	**Long-Term Flexibility**	**Integration ability**
Mathematical Techniques	0	-	0	-
Dispatching Rules	-	0	+	+
Simulation-Based Dispatching Rules	0	0	0	-
Knowledge-Based systems	+	0	-	0
Search Heuristics	+	0	0	0

Dispatching rules are associated with poor efficiency due to their consideration of the local queue at the machine level. In addition, they are typically oriented to single resources, especially to machines. This implies their limited capability to provide a comprehensive scope of the shop floor, which leads to poor utilization of the resources. Dispatching rules are distinguished by their good reactivity since most of them are associated with low computational complexity. However, they provide no support for the sophisticated handling of disturbances, which is only possible if they are to be combined with other techniques. This justifies their incapability to fulfill the short-term flexibility requirement. The simplicity of realizing dispatching rules results in relatively high long-term flexibility. In addition, most of these rules are based on simple data about the jobs in the local queue, and hence are easily integrated in dynamic environments.

Combining dispatching rules with a simulation technique certainly leads to better efficiency at the global level. On the other hand, the short-term flexibility is evidently deteriorated with respect to the reactivity. However, this deterioration is balanced by the enhanced ability to handle disturbances. Similarly, the long-term flexibility is weakened due to the need to update the simulation module whenever the system is to be extended. Moreover, the integration ability of scheduling systems applying simulation-based dispatching rules is hindered by the overwhelming data required from the environment to carry out the simulation.

Knowledge-based approaches are always centralized. Their global centralized view of the shop floor allows them to generate high quality schedules, which is reflected in the high efficiency evaluation assigned to them. A knowledge base typically requires considerable effort to be built and maintained, which is hardly achievable in dynamic environments [OuPe08]. This implies a high computational complexity and accordingly low reactivity. On the other hand, knowledge-based systems provide good support for handling disturbances. That is why their short-term flexibility is evaluated as neutral despite their poor reactivity. However, their shortcoming becomes more evident with respect to the long-term flexibility due to the difficulty of applying

changes to the knowledge base. Mostly, the effort of integrating knowledge-based systems corresponds to the initial integration. During operation, data can flow from the control to the knowledge-based systems, which typically depend in their decisions on a reasonably sized set of parameters. However, the integration ability requirement is only partially satisfied due to the centralized structure of the knowledge-based approach which hinders the integration of the scheduling system into large-scale FMSs.

The efficiency of search heuristics is good since the generation of a near-optimal solution is usually guaranteed. Search heuristics are generally associated with good reactivity. Heuristics are based on the principle of minimizing the computational complexity. Therefore, their reactivity is relatively high. However, they have to be complemented with more sophisticated techniques to be able to adapt the generated schedule to unforeseen events. They are associated with relatively low long-term flexibility due to their inherently centralized structure. However, the effort required to extend search heuristics is low compared to that of extending knowledge bases. The integration ability of search heuristics is partially satisfied.

In summary, no approach satisfies all the requirements. However, the knowledge-based and search heuristic approaches are characterized with better performance in terms of efficiency. Out of these two approaches, the search heuristics approach seems to partially satisfy the rest of the requirements.

3.4 Concluding Remarks

Having reviewed the conventional approaches for FMS scheduling from three different perspectives, the following points can be concluded:

- Scheduling for FMSs has to provide support for a comprehensive scope of the available resources and accommodate for not only machine-centric scheduling but also for integrated scheduling. Supporting integrated scheduling does not mean that machine-centric scheduling, or production scheduling, should be completely abstained. In some FMS configurations, production scheduling can be sufficient due to the minor role played by the transportation time in determining the total time required for the production flow. On the other hand, a dynamic scheduling approach is necessary to allow for the generation of schedules under consideration of the environmental dynamics. The survey of literature has shown that the agent-based approach is the most promising approach in this respect. However, its wide-scale adoption is hindered by the need to complement it with suitable architectures and scheduling techniques.

- The possible system architectures have been surveyed out of their decomposition art and organizational structures. It can be concluded that the physical decomposition provides a natural mapping for the shop floors of FMSs, in that resources involved in scheduling can be

represented as computational entities. This decomposition art enhances the extensibility of the system and facilitates the negotiation-based scheduling, which suits the agent-based approach advocated previously. Similarly, the existing organizational structures have been surveyed and evaluated. Out of this evaluation, the heterarchical structure was found to outperform the other structures.

- The conventional scheduling techniques have been surveyed and assessed. The evaluation resulted in revealing that search heuristics provide the best relative performance, when the requirement on efficiency is emphasized.

These results have guided the basic decisions of the current research, represented in:

- Adopting an agent-based approach for conceptualizing flexible scheduling
- Applying search heuristics for the optimization of the generated schedules

In this way, the proposed concept combines the flexibility of agents with the optimization support of the search heuristics. Before getting into the details of the developed concept, these basic decisions are discussed in more details, in the next chapter, to provide more insight into the rationale behind these decisions and present the foundational knowledge, required for conceiving the applied techniques.

4 Basic Decisions of the Conception

This chapter investigates the two basic decisions, derived in the previous chapter in more details. The first decision corresponds to the adoption of the agent-based paradigm to exhibit the required flexibility. To give an insight into this decision, the main concepts of agency are reviewed and the strengths of the agent-based paradigm with respect to FMS scheduling are emphasized. Following this discussion, the second decision, concerning the application of search heuristics for the schedule optimization, is addressed. After an introduction to their underlying principle and their general procedure, the suitability of search heuristics to FMS scheduling is explained. Having elaborated on these two main decisions, the chapter concludes with an overview on the supported scheduling modes. Different scheduling modes have to be supported to account for the different environmental conditions and the different scheduling scopes. This overview acts as a roadmap of the proposed concept, since it given an insight into the chosen structure of the following chapters, explaining the concept, based on the classification of the scheduling modes.

4.1 Adopting an Agent-Based Approach

The agent-based approach is strongly believed to be very promising in tackling the complexity of manufacturing scheduling based on its inherent flexibility [Leit08] [Shen02b] [SWH06]. The advantages of agents become even more apparent when it comes to FMS scheduling, for which flexibility is a crucial and decisive concept. The underlying principle of agent-based scheduling is to decompose the problem into self-contained entities and to represent these entities by autonomous agents. Schedules are generated based on negotiations among these autonomous entities. The autonomy and interactivity of agents provide very good support for tackling the inherent complexity of the scheduling problem and its interrelationships with the various decision-making manufacturing tasks. In what follows, the basic concept of agency is reviewed and the decision of adopting an agent-based approach for FMS scheduling is justified, by discussing the suitability of agents to the stated requirements.

4.1.1 Basics of the Agent-Based Paradigm

The autonomous agents' research has emerged as a branch of Artificial Intelligence during the eighties. It is referred to as behavioral-based AI as opposed to the mainstream knowledge-based AI [MaKo93]. The word "agent" – from the Latin word "agäns": to act – is defined as the producer of an effect [Toko95]. Researchers have been proposing several definitions to capture the nature of autonomous agents. One of these definitions states that an agent is an encapsulated entity with defined goals. An agent endeavors to accomplish its goals through its autonomous

behavior and in doing so it interacts continuously with its environment as well as with other agents [WGU03].

Whatever the variation in the proposed definitions is, there is a kind of consensus among researchers on the necessary attributes that are distinguishing agents from other software entities. In what follows, a list of these attributes is given along with a brief clarification of each.

- Encapsulation: Similar to objects, agents encapsulate their state and identity by providing an abstract interface. In addition, agents encapsulate their internal behavior [WGU03].
- Goal orientation: An agent works according to determined goals that are either specified statically at development time or delegated from user dynamically [FrGr96] [Badr07].
- Autonomy: While objects encapsulate their state and passive behavior which is exhibited by function calls, agents control their active behavior that stems from the action selection freedom, which an agent possesses [WGU03].
- Reactivity: An agent is reactive in that it possesses the ability to sense changes in the environment and react to it accordingly [FrGr96].
- Proactivity: In addition to being reactive, agents are proactive in that they take the initiative and exhibit goal-oriented behavior without being externally triggered to do so [FrGr96].
- Social ability (communication ability): Agents have the ability to cooperate in fulfilling the common objective of the whole system. This cooperation occurs through different methods of cooperation like task delegation, negotiation and team work [Dam03].

In addition, multi-agent systems support openness by allowing new agents to register and integrate themselves dynamically. This feature aids in scalability and extensibility.

4.1.2 Suitability of Agents to FMS Scheduling

In this section, the benefits of agents with respect to the individual requirements, derived in chapter 2, are discussed.

4.1.2.1 Benefits of Agents Regarding Efficiency

It can be argued that agent-based scheduling leads to reduced efficiency because of two main reasons. First, the myopic view of the individual agents obscures reaching an optimal schedule at the global level. Second, the attempt to complement agents with classical optimization techniques from the scheduling theory is hindered by the fact that these techniques are inherently centralized and it is not straightforward to adopt them in the decentralized agent-based architectures. Paradoxically, we argue that agents possess features that are associated with the following gains, which facilitate the fulfillment of the efficiency requirement.

Multi-Objective Integrated Optimization

The goal-orientation of agents facilitates the multi-objective optimization by decomposing the problem into entities, each of which works on achieving its predefined goals. The autonomy and interactivity of agents can drastically reduce the complexity associated with the integrated scheduling. In other words, the complexity can be tackled by decomposing the problem into sub-problems and delegating them to agents that can work in parallel to achieve their local goals. By making these agents interact with each other, the generated schedule can be optimized at the global level.

Utilization of Shop Floor Flexibility

The machine flexibility implies that the setup of the machines changes over time and their supported operations change accordingly. This change in the capabilities over time hinders the conventional approaches from considering the collective capabilities while optimizing the schedule. However, the flexibility of the agent-based communication provides an intuitive solution for addressing this problem.

4.1.2.2 Benefits of Agents Regarding Short-Term Flexibility

In agent-based scheduling, decisions can be made close to the corresponding physical entities which enhances reactivity and adaptability.

Reactivity

The agent-based approach is deemed to yield low computational complexity compared to other approaches especially the centralized approaches. This reduction in complexity is attributed to the decentralization which allows for the parallel execution [BMG08]. In addition, the decomposition into agents allows for limiting the execution of the schedule generation and update to the concerned agents. However, special care has to be given to the communication complexity associated with the agent-based scheduling. This can be achieved by managing the agent-based modeling and communication protocols.

Robustness

By fulfilling the requirement on the utilization of shop floor flexibility, resource failures can be handled dynamically through the collective optimization of resources. The breakdown of a resource can be captured at the decentralized level and can be automatically handled by contacting agents responsible for other working resources with the same capabilities.

4.1.2.3 Benefits of Agents Regarding Long-Term Flexibility

Multi-agent systems support openness by allowing new agents to register and integrate themselves dynmically. This is regarded as one of the most appealing features of agents due to ease of incorporating long-term changes with minimal programming effort.

Extensibility and Modifiability

Decomposing scheduling into agents with a one-to-one correspondence to the physical resources facilitates the addition of new resources to a great extent. In addition, agent-based physical decomposition can aid in enhancing the modifiability of physical and logical entities represented by agents because of their associated abstraction and high cohesion.

Configurability

The encapsulation and goal-oriented decomposition of agents facilitates the configurability of optimization goals at run time. A change to an optimization objective can be realized at run time by simply informing the agent or agents responsible for this objective.

4.1.2.4 Benefits of Agents Regarding the Integration Ability

By definition, agents are situated in their environments and possess the capabilities of sensing and reacting. This is very promising for integrating agents in the dynamic FMS scheduling environment.

Real-Time Monitoring

The difficulty of monitoring distributed resources can be tackled by delegating this responsibility to agents. This ensures that the process takes place in parallel which facilitates capturing relevant events in real-time.

Transparency

Realizing scheduling as agents improves the transparency to the decision makers. The current status of operation can be easily grasped by dynamically communicating with the agent in charge. The operators are relieved from having to take care of the low level details entailed in the conventional scheduling systems. This is attributed to the agent-oriented abstractions, which allow for the interaction with agents at the knowledge level [Jenn01].

4.2 Applying Search Heuristics for Schedule Optimization

Despite their appealing benefits, the adoption of agents to FMS scheduling has been hindered mainly because of their lack of optimization support. For agents to achieve the required efficiency of the generated schedule, they have to accommodate for more sophisticated

scheduling and optimization techniques [Shen02b]. To tackle this problem, search heuristics are applied to equip agents with the required optimization support. The selection of search heuristics resulted from the analytical survey of scheduling techniques, conducted in chapter 3. Heuristic-based search tackles the inherent complexity of the scheduling problem by its capability of finding a good solution within reasonable time limits. Although they do not guarantee the localization of optimal solutions, heuristic-based search techniques are promising when a near-optimal solution is sufficient. Therefore, they suit the FMS scheduling problem quite well due to their balanced performance with respect to both reactivity and efficiency.

While genetic algorithms are applied for the optimization of production schedules, the A* algorithm is applied for the optimization of transportation schedules. In this section, these two techniques are described and the decision to apply them in the proposed agent-based scheduling is justified.

4.2.1 Applying Genetic Algorithms for Production Scheduling

Genetic algorithms have been successfully applied to the optimization of manufacturing scheduling [GeCh97] [Shen02a]. Before discussing the gained advantages of their application for production scheduling, an introduction to the underlying principle and main structure of GAs is presented.

4.2.1.1 Overview on Genetic Algorithms

Genetic algorithms are stochastic multi-heuristic search techniques, whose development is credited to Goldberg. They are inspired by the biological evolution and work by iteratively evolving solutions from previous ones until a feasible solution with reasonable quality is found. The procedure of genetic algorithms is described as follows [GeCh97]:

```
Procedure: Genetic Algorithms
begin
   t ← 0
   initialize P(t);
   evaluate P(t);
   while (not terminal condition) do
      recombine P(t) to yield C(t);
      evaluate C(t);
      select P(t+1) from P(t) and C(t);
      t ← t + 1
   end
end
```

The algorithm starts by creating an initial set of solution candidates, referred to as the initial population *P(0)*. This initial population consists of chromosomes that are usually generated at random. Each chromosome encodes a solution candidate based on a given representation, such as the binary representation. Having generated the initial population, every chromosome is

evaluated by calculating the value of a fitness function. The fitness function provides estimation to the fulfillment of the chromosome to the optimization objectives. Chromosomes evolve through successive iterations called generations. In creating a new generation, chromosomes from the current population are either combined using a crossover operator or modified by a mutation operator. Out of the parents P(t) and the offspring C(t), chromosomes with a fixed number, amounting to the population size, are selected based on their fitness values. This process continues until a predefined terminal condition is satisfied, such as reaching a given number of generations. Upon reaching the final condition, the best chromosome out of the final population is selected as the best solution candidate found so far.

Genetic algorithms provide a general-purpose evolution-based optimization procedure which needs to be adapted to the specific application. In applying GAs to a specific problem, a chromosome representation and a fitness function have to be specified. In addition, GA-parameters, like the population size, have to be configured.

4.2.1.2 Suitability of GAs to Production Scheduling

Genetic algorithms have been successfully applied to the job shop scheduling problem [GeCh97] [YaNa97] [MaBi04], which resembles to a large extent the production scheduling of FMSs, as illustrated in section 2.2.3. They are believed to outperform other meta-heuristic search methods, such as tabu-search and simulated annealing, by their resistance to premature convergence to local optima [LGP97]. The performance of GAs in solving well-known benchmark job shop problems is surveyed in [YaNa97] and [GeCh97]. Experimental results prove that GAs are very promising in converging to the optimal schedule in reasonable time.

The extension of agents with GAs poses the challenge of adapting the centralized structure of GAs to the decentralized structure of the agent-based scheduling. Decentralizing GAs to be incorporated into the proposed agent-based architecture has to account for the goal-orientation and the myopic nature of agents. This issue is addressed in details in chapter 7.

4.2.2 Applying A* for Transportation Scheduling

The complexity of the transportation scheduling problem lies in the routing sub-problem which belongs to the class of pathfinding problems. For solving this problem with reasonable computational complexity, the A* algorithm is applied. A* is a widely-used heuristic-based pathfinding algorithm, which represents an extension to the Dijkstra's algorithm. The Dijkstra’s algorithm is guaranteed to find the optimal path in an efficient way, which justifies its applicability in different areas such as packet routing and car navigation [Dies97].

Both the A* and the Dijkstra's algorithms work by traversing a weighted graph representation of the problem at hand, as the graph depicted in Figure 4.1. In the AGV routing problem, vertices of the graph represent machines, buffers or load/ unload stations. Connections between vertices

correspond to paths linking these physical resources. Every connection is associated with a direction and a cost. The cost can be directly given or derived, by for example estimating the time taken to travel the path. In the example depicted in Figure 4.1, three different routes are possible to travel from A to E, namely (A,F,E), (A,C,F,E), and (A,C,B,D,E). In searching for the best route from A to E, the Dijkstra's algorithm traverses the neighbors of the active vertex, initially the source vertex A, taking their associated cost into consideration. It returns the optimal route, which is (A,C,F,E).

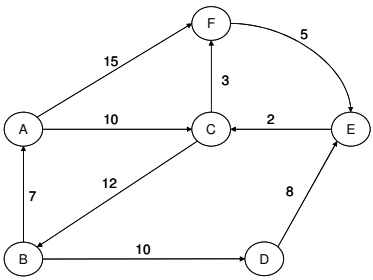


Figure 4.1: A sample graph representation of a shop floor

The A* algorithm extends the Dijkstra's algorithm by traversing the neighboring vertices, under consideration of both the actual distance of the considered route along with a heuristic, which usually estimates the additional expected distance from a considered vertex up to the destination. In this way, the A* algorithm finds a route with low cost much faster than the Dijkstra's algorithm [RuNo04]. The decision to apply A* for optimizing the AGV routes resulted from an analytical study of conventional pathfinding methods, that was conducted in the course of a graduate project at the IAS [Schm09].

4.3 Supporting Different Scheduling Modes

The integrated nature of scheduling within planning and shop floor control coupled with the stochastic and dynamic nature of the environmental conditions of FMSs imposes flexibility requirements on scheduling, as detailed in section 2.3. This required flexibility necessitates the adaptation of scheduling to the current environmental configuration. In other words, different scheduling types, also referred to as scheduling modes, are envisioned. To derive the necessary scheduling modes, the external factors affecting the proposed FMS scheduling are analyzed in the context diagram depicted in Figure 4.2. The dynamic data that flow from planning and control to scheduling correspond to the required scheduling configuration, the generation and modification requests and the current control status.

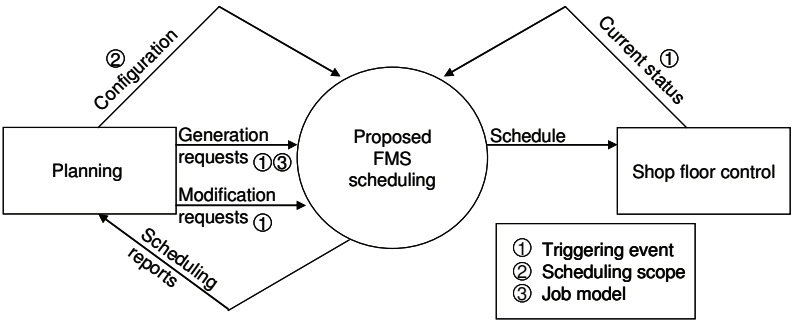


Figure 4.2: Context diagram of the proposed FMS scheduling

Out of these data flows, three shaping factors in the scheduling mode selection can be derived, as depicted in Figure 4.2. The first factor is the triggering event, which causes a schedule to be generated or repaired. Scheduling is triggered by either a normal schedule generation request, represented in the arrival of a set of planned jobs, a modification request, causing changes to the scheduled jobs, or a change in the current status of the operation, such as a failure in one of the resources. The second factor is the scheduling scope, which is selected based on the configuration parameters. The third factor is the job model, which relates to the number of job requests released from planning to scheduling at a given point in time. In what follows, the effect of these factors on scheduling is explored to derive the required scheduling modes, which are illustrated in Figure 4.3.

4.3.1 Scheduling Modes Accounting for Different Triggering Events

As illustrated in Figure 4.2, triggering events are related to three dynamic flows: requests received from planning for schedule generation and schedule modification and the current status, captured from the shop floor control. Schedule generation requests are distinguished from schedule modification requests in that they cause no modification to the already scheduled jobs. Schedule modification requests include the cancellation or delay of already scheduled jobs. The current status of the shop floor control triggers modifying the schedule in case a problem in one of the resources is detected. These contingencies occurring during control together with modification requests received from planning are referred to as disturbances. While receiving a generation request triggers the schedule generation mode, the occurrence of a disturbance triggers the schedule repair mode, as shown in Figure 4.3. These two modes differ in their objectives and the applied methods and are thus handled separately in the thesis.

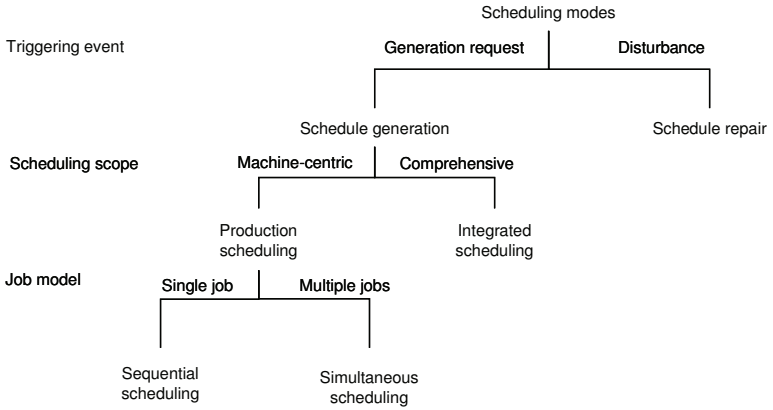


Figure 4.3: Taxonomy of the supported scheduling modes

4.3.2 Scheduling Modes Accounting for Different Scheduling Scopes

In chapter 3, two different scopes have been surveyed. Out of this survey, it has been concluded that both the machine-centric and the comprehensive scopes are necessary and have to be considered by FMS scheduling. Selecting one of these scopes is related to the physical layout and the size of the underlying shop floor. For relatively small FMSs, the material handling time could be disregarded in comparison to the processing time and hence a machine-centric scope would be sufficient. In case the FMS has to be expanded, a decision can be made to widen the scheduling scope and take into consideration not only the machines but also other resources employed for material handling. Based on the selection of the scope, schedule generation can be classified into production scheduling and integrated scheduling, corresponding to the machine-centric and the comprehensive scopes respectively (See Figure 4.3). In integrated scheduling, production scheduling is integrated with transportation scheduling.

4.3.3 Scheduling Modes Accounting for Different Job Models

A distinction is made according to whether jobs arrive to scheduling one by one or collectively, in the form of multiple jobs at a time. The current job model triggers either the sequential or the simultaneous scheduling modes, corresponding to the single job and the multiple job cases respectively (See Figure 4.3). It is worth noting that this classification applies only to the production scheduling mode, since the integrated scheduling mode is supported with only the case of jobs arriving one by one. This assumption is justified by the complexity of the integrated scheduling problem. Supporting the simultaneous scheduling mode for integrated scheduling is envisioned as future work.

This chapter has highlighted the basic decisions associated with the proposed concept and has laid the foundation necessary for comprehending the applied techniques. In addition, the supported scheduling modes were derived and explained. The subsequent chapters, addressing the concept, are structured as follows. Chapter 5 develops the system architecture which applies for all the scheduling modes. Chapters 6 and 7 deals with the schedule generation mode, along with its corresponding lower-level modes. The schedule repair mode is thoroughly discussed in chapter 8.

5 The Proposed Agent-Based Scheduling Architecture

The analytical survey, conducted in chapter 3, resulted in the decision to adopt an agent-based architecture with physical decomposition and heterarchical organizational structure. This chapter is devoted to deriving and explaining the proposed architecture. First, an overview on the proposed architecture is presented. Second, a static model capturing the FMS-specific details, that are relevant to scheduling, is developed. Third, the organizational structure of the agent-based architecture is worked out by focusing on the different layers of abstraction. Fourth, the agent types at the different layers are identified and described. Finally, the internal agent architecture is illustrated.

5.1 Overview on the Proposed Architecture

The basic idea of the proposed architecture is to provide generic support for FMS scheduling based on the separation of the FMS specific details, which are relevant to scheduling, from the scheduling-generic functionalities. The former is represented as a centralized model, capturing these specific details, as will be explained in the next section. The latter is represented by agents, which are integrated into planning and shop floor control, as depicted in Figure 5.1. Generic agent types, required for FMS scheduling, are defined. These agent types can be parameterized and instantiated at run time without having to rewrite the scheduling software when deploying it on a different FMS or when introducing structural changes to the existing FMS.

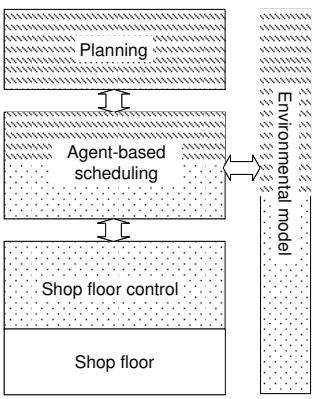


Figure 5.1: Overview on the proposed FMS scheduling architecture

To account for the required integration, the scheduling environment is modeled in the proposed agent-based scheduling, as captured in Figure 5.1. This modeling is made possible by representing logical and physical entities from planning and shop floor control as agents and allowing these agents to capture the dynamic status of their corresponding entities. In addition, this representation tackles the complexity of the FMS scheduling problem by decomposing it into sub-tasks and assigning them to the concerned agents [Badr08]. Every agent endeavors to reach its goal by performing its assigned sub-task. Through the interaction of agents with the environment and with each other, schedules are generated under consideration of the overall scheduling goals. The schedule itself is maintained from the different points of view of the involved agents. In other words, there is no central schedule and accordingly no central entity responsible for managing the schedule.

5.2 Modeling the Environment

As previously mentioned in chapter 2, scheduling is concerned with the real-time planning of the production flow. This production flow consists of five steps, whose execution depends on resource-related factors that have to be taken into consideration. These factors are classified into static and dynamic factors [BaGö09b]. Static factors correspond to the characteristics of the physical resources and the planned part types and their interdependencies. Dynamic factors, on the other hand, are related to the current status of these resources. The objective of the environmental model is to abstract the proposed agent-based scheduling from the FMS specific details by capturing the static factors, influencing scheduling. These factors are identified in this section by analyzing the steps constituting the flow of production.

Table 5.1: Derivation of the static influencing factors

Production step	**Influencing resources**	**Static influencing factors**
Loading	Material	Material type for a given part type
Transportation	AGV	Capacity and speed of AGV Physical layout of the shop floor
Temporary storage	Local buffer, work-in-progress buffer	Storage space
Processing	Machine, tools	Operations required by a given part type Operations supported by a given machine Setup time required for a given part type Tools required for a given operation and a given part type
Unloading	Store	Storage space

Table 5.1 lists the steps of the production flow and derives for each step the influencing resources and the static influencing factors. For a schedule to be generated for a given job, corresponding to the production of a certain quantity of a certain part type, the availability of the

material required for this part type has to be ensured. Different material types are normally associated with different part types. Therefore, the associations between part types and material is one of the static factors that are relevant to scheduling. For scheduling the transportation of material or workpieces, the capacity and speed of AGVs are necessary to estimate the transportation time. However, this has to be done under consideration of the physical layout of the shop floor, which has to be modeled as well. The physical layout corresponds to the connections among the physical resources along with their associated distances.

As illustrated in Table 5.1, the temporary storage step is associated with either the local buffer of the machines or with a shared work-in-progress buffer. In both cases, the storage space of the buffer is relevant to scheduling, as a limiting factor for the number of workpieces that can be executed at a given point in time. For the processing step, a number of static factors have to be taken into consideration. First, the operations required for processing a given part type are required to allocate machines to jobs. Due to the flexibility of the machines in supporting operations, the dynamic switching of a machine from an operation to another is possible. The potential operations supported by every machine are constrained by the physical specifications of the machines and are relevant to dynamic scheduling. A direct factor influencing the time associated with the processing of a certain job at a given machine is the setup time required for the part type corresponding to the job. Finally, the interrelationships among tools, part types and operations are relevant to dynamic scheduling, since the breakdown of a given tool has to be handled by scheduling. For handling such disturbance, the part types and operations affected by the tool breakdown have to be made available. The final step of the production flow is the unloading step which requires keeping track of the storage space, similar to the loading step.

These derived factors are captured in the environmental model by distinguishing between factors related to the processing and the material handling systems. Factors related to the processing system are modeled by identifying entities and associations among them. Factors associated with the material handling system are specified in relation to the physical layout and described in the form of an XML schematic description that can be dynamically interpreted by the scheduling system.

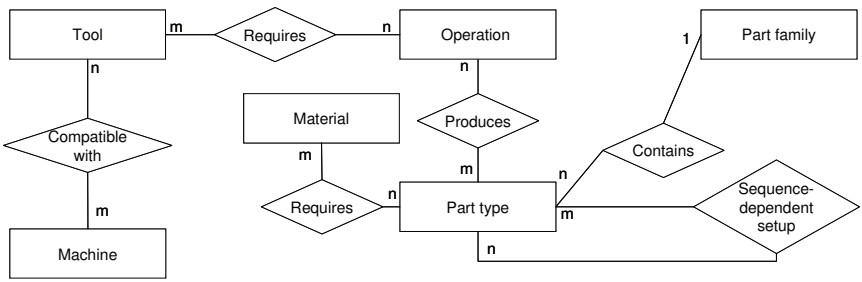


Figure 5.2: An entity relationship model of the processing subsystem of a typical FMS

Figure 5.2 depicts the entities identified for modeling the static factors of the processing system along with the associations among them. As illustrated in Figure 5.2, the derived influencing factors are modeled by identifying six entities: machine, operation, tool, material, part type and part family. It is worth noting that the setup time is modeled in two different forms to account for the two different types of setup time namely, the sequence-dependent and the sequence-independent setup time. In case of FMSs with sequence-independent setup time, the required major setup time is considered as an attribute describing the part family entity. Likewise, minor setup time values are regarded as attributes for the part type entity. However, in case of FMSs with sequence-dependent setup times, the setup time is regarded as an attribute of the relation between a given part type and the one preceding it on a certain machine, as captured in Figure 5.2. It is worth noting that the considered FMSs are assumed to be fully automated, which is the reason behind disregarding the human resource from the environmental model.

5.3 Layers of the Proposed Architecture

By definition, scheduling is concerned with the allocation of available resources to the planned jobs. This motivated the introduction of two layers, one for jobs and the other for resources. While the former corresponds to planning, the latter is related to shop floor control. To capitalize on the components flexibility of FMSs resulting from the grouping, described in chapter 2, resources are grouped according to the service they currently provide. This results in introducing a layer for representing the services provided by resources, above the resource layer, as in Figure 5.3. This added layer accounts for the collective capabilities of the available resources while generating and updating schedules. Similarly, jobs are grouped based on their shared constraints, such as a shared deadline for a shared delivery destination. This grouping results in the introduction of a layer above the jobs layer, as in Figure 5.3. The introduction of the job group layer aims at accounting for the individual requirements of the different customers, corresponding to the different delivery destinations, while generating schedules.

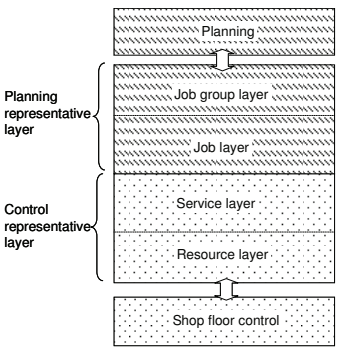


Figure 5.3: Layers of the proposed architecture

5.4 Deriving Agent Types

The main principle underlying the proposed agent-based dynamic scheduling is to tackle the inherent complexity of FMS scheduling by decomposing the problem into sub-problems and assigning them to the concerned agents. Each agent endeavors to solve its assigned sub-problem out of its limited knowledge and by aiming at the accomplishment of its local objective. In this section, the agent types at the four layers of the proposed architecture are derived.

5.4.1 Agent Types at the Resource Layer

Resource agents represent resources affecting the flow of production, illustrated in Figure 2.6. Since the number of resources affecting the flow of production, such as tools, AGVs and buffers, is large, a one-to-one representation of agents to resources would yield a large number of agents and thus high communication complexity at run time. This is dealt with by classifying resources into passive and active resources. As their name suggests, passive resources are not assigned any responsibility in scheduling and hence do not have to be represented by agents. However, the availability of these passive resources is necessary for following the schedule. In other words, their status has to be monitored in order to react to any potential failures in them, by repairing the schedule accordingly. Therefore, active resources are represented by agents, which are assigned the responsibility of capturing the current status of their corresponding active resources along with the current status of the passive resources currently residing in them. A prerequisite for this distinction is to ensure that a passive resource has to be physically located at an active resource at any point in time.

Table 5.2: Derivation of the influencing resources with the corresponding locations

Production step	Influencing resources	Influencing location
Loading	Material	Material store
Transportation	AGV	AGV
Temporary storage	Local buffer, work-in-process buffer	Machine, work-in-process buffer
Processing	Machine, tools	Machine
Unloading	Store	Final product store

Identifying active resources is based on an analysis of the flow of production, depicted in Figure 2.6. This flow is analyzed in Table 5.2 by considering the production steps and deriving for each step the influencing resources and the influencing locations, corresponding to the active resources. As presented in Table 5.2, five influencing locations can be derived from the five different production steps. Based on these influencing locations, the required agent types along

with their responsibilities in scheduling are derived in Table 5.3. This results in four agent types representing machines, AGVs, buffers, including both work-in-process buffers and final product stores and material allocators, corresponding to the material store. Based on these four agent types, a comprehensive coverage of the current status of all the relevant resources is supported.

As illustrated in Table 5.3, the main responsibility of machine agents is to solve the sequencing sub-problem of machine scheduling. Likewise, AGV agents are responsible for solving the routing sub-problem of the transportation scheduling. The material allocator and the material store agents carry auxiliary responsibilities in transportation scheduling by allocating material and store space, respectively. They aid in generating a transportation schedule under consideration of the availability of the material and the store space resources.

Table 5.3: Derivation of agent types and the associated responsibilities at the resource layer

Influencing location	**Agent type**	**Scheduling responsibility**
Material store	Material allocator agent	Allocating material to jobs
AGV	AGV agent	Solving the routing sub-problem
Work-in-process buffer, final product store	Buffer agent	Allocating store space to jobs
Machine	Machine agent	Solving the sequencing sub-problem

5.4.2 Agent Types at the Service Layer

Agents at this layer represent services supported by a group of resources. As previously explained in section 2.2.2, the steps of the production flow are classified into processing steps and material handling steps. These two classifications correspond to services provided by the resources to the jobs. In other words, a resource can provide a job with either a processing service, by performing an operation, or a transportation service, by moving workpieces from a source to a destination. Transportation services include other related tasks such as the temporary storage of workpieces. Accordingly, two main groupings of resources are possible: grouping machines based on their operations and grouping material handling resources based on their transportation service. Material handling resources provide different transportation services if they are located in different partitions at the shop floor, referred to as flexible manufacturing cells (FMCs).

This results in identifying two main agent types: operation agents for representing processing services and transportation agents for representing material handling services. A sample FMS consisting of one FMC is depicted in Figure 5.4. For this shop floor, three service agents are required: two operation agents for representing the shared operations of the CNCs, determined by their current setup, and a transportation agent for managing the material handling resources.

The objective of operation agents is to improve the utilization of the machines currently performing the operation in concern. Similarly, transportation agents aim at minimizing the transportation time required for incoming job requests by enhancing the utilization of the corresponding material handling resources. In other words, operation agents cooperate with their machine agents to optimize production schedules and transportation agents cooperate with their material handling resources to optimize transportation schedules.

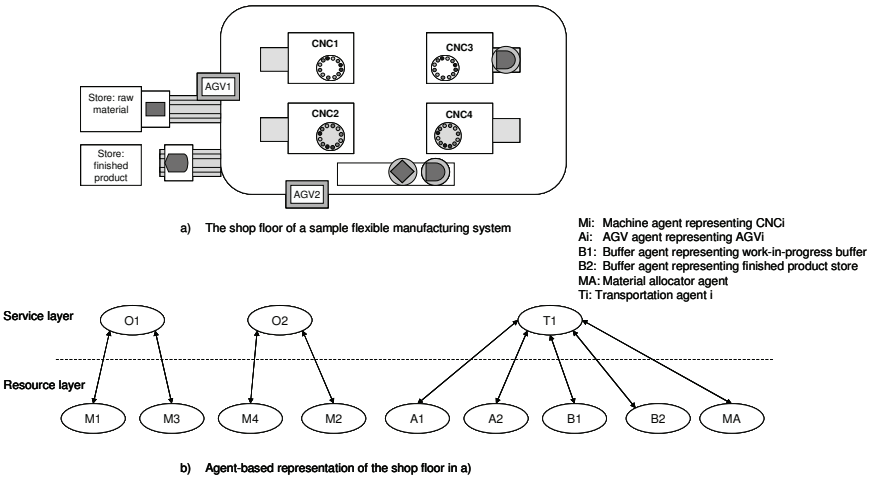


Figure 5.4: Agent-based representation of a sample FMS shop floor

5.4.3 Agent Types at the Job Layer

In representing jobs, a job agent type is defined. Job agents represent planned jobs, which are described by their part types, quantities, priorities, release dates and possibly deadlines. The objective of a job agent is to get the best allocation of the required resources over time. This implies that its responsibility in scheduling is to evaluate the offers provided to it by the operation and transportation agents. The evaluation is based on the total time offered for executing the job or the deviation from the deadline. Job agents have to make sure that the offered schedules are feasible concerning any precedence constraints, that might be involved in their work plans.

5.4.4 Agent Types at the Job Group Layer

At this layer, a job group agent type is defined. This agent type aims at achieving an allocation of its jobs that leads to satisfying the common constraints. Grouping jobs can be based on either their part types or their delivery destination, also referred to as customers. The decision on the

grouping art is taken at the planning phase and depends on the strategy of the FMS. If production requirements are specified based on the capacity of the system, it makes sense to group jobs based on shared part types and consider the predefined production requirements of every part family in the group agent representing this family. On the other hand, if the system produces based on customer requests, job group agents represent the set of jobs for a given customer. In summary, while the former way of grouping accounts for the system view the latter accounts for the customer view. In this work, the latter view is adopted since it accords with the open shop model of the FMS, explained in section 2.2.1.1.

5.5 Internal Agent Architecture

Conventional agent architectures can be classified into reactive architectures, deliberative architectures and interactive architectures [Müll98]. Each of these architecture types stresses one property of the agent behavior. Reactive architectures emphasize the reactivity aspect of agents by basing their decisions mainly on sensory input. Deliberative architectures possess a more sophisticated symbolic representation of the encompassing environment and stress the proactive goal-oriented aspect at the cost of reactivity. Interactive architectures are concerned with the interaction among agents and lay their focus on incorporating communication mechanisms and protocols. Motivated by the shortcomings of these approaches, hybrid architectures have been proposed. However, there has been no agreed-upon architecture and the choice of an agent-based architecture has remained a solution-specific issue.

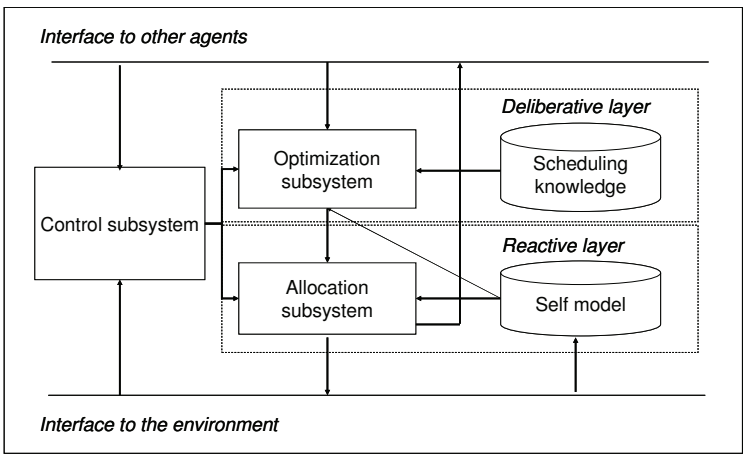


Figure 5.5: The internal agent architecture

For the purpose of this research work, an architecture inspired by the requirements, derived in section 2.3, has been developed. This architecture accounts for the required combination of

flexibility with efficiency. As illustrated in Figure 5.5, the proposed architecture consists of two layers: the reactive layer and the deliberative layer. While the reactive layer is intended for the quick reaction to the environmental dynamics, the deliberative layer is responsible for optimizing the quality of the agent's decisions. The two layers are complemented by a control subsystem which undertakes the activation of the suitable layer based on the current status of the environment and the current configuration of the agent behavior. In what follows, the individual components of the architecture are described in more details.

5.5.1 Interface to the Environment

As previously described in section 2.2.1, FMS scheduling is highly interdependent on its surrounding environment. A distinction is made between static and dynamic environmental factors. While the former can be statically represented in the environmental model, the latter have to be made available to scheduling dynamically. Accordingly, the proposed scheduling approach emphasizes the integration of scheduling into its environment by providing interfacing channels linking scheduling with the environmental model as well as with the planning and shop floor functionalities.

The interface to the environment provides agents with sensing and acting capabilities. The environment corresponds to the planning layer, the shop floor control layer and the environmental model. The interface to the environment involves all the primitives required for providing accessibility to the three environmental entities.

5.5.1.1 Interaction with Planning

The objective of this interface is threefold: enabling configurability, providing transparency and making planned requirements available to the concerned agents. While the configurability of agents corresponds to the long-term flexibility, the transparency enhances the integration ability. For satisfying the configurability, parameters are defined for controlling the behavior of agents. These parameters correspond to the scheduling scope and to optimization-related parameters, which will be detailed in chapter 7.

Decisions taken by agents are made transparent to the planning layer. These decisions basically involve the state of the schedule generation under the different conditions. This allows for evaluating the performance of the agent-based scheduling under the different conditions. The state of the schedule generation is proactively archived by the agents themselves and can be invoked upon request. The schedule generation state includes, in addition to the eventually generated schedule, the partial schedules attained in the course of the schedule generation. This can support the management in taking future decisions regarding the configuration of scheduling and thus enhances the visibility into the agent-based behavior.

Finally, the interface with the planning layer supports communication channels for conveying both generation requests as well as modification requests. In general, the communication channels used for interfacing with the planning layer are asynchronous channels. This can be realized based on the message passing mechanism supported by agents.

5.5.1.2 Interaction with Shop Floor Control

The interaction with shop floor control is necessary for interleaving scheduling with schedule execution or realizing the desired online scheduling. The requirement which is focused on by this interface is the requirement on the real-time monitoring, which is necessary for satisfying the integration, explained in section 2.3. Therefore, this interface is provided to agents at the resource layer in form of communication channels for equipping them with the capability of capturing the dynamics of the control and transmitting the schedule to the concerned resources for execution. The dynamics of the shop floor that are captured by scheduling include events like the termination of a scheduled job, the current load of resources and possible resource failures. It is worth noting that the used communication channels are intended for the integration with a simulated control, in the current research work.

5.5.1.3 Accessing the Environmental Model

The environmental model constitutes part of the environment and thus has to be made accessible to agents. The access to this model is made available based on the object-relational mapping concept. In other words, database objects are mapped into programming objects which can be accessed by agents without having to get into the low details of the applied database. In addition to the abstraction from the low-level details of relational databases, object-relational mapping allows for exploiting the merits of the object-oriented paradigm, such as encapsulation, inheritance and polymorphism.

The provided interface supports bi-directional access to the environmental model. Data stored in the model have to be retrieved whenever static data are required. For example, the operations required for manufacturing a given part type have to be retrieved, whenever a job associated with this part type is requested. In addition, access to the environmental model can take the form of temporal archiving, which justifies the write access granted to the agents and symbolized by the arrow directed to the environmental model in Figure 5.1. An example for this is to save the relevant data of all jobs to enable other agents to retrieve job attributes based on a given identifier. The reason for this is mainly to minimize the communication load among the agents.

5.5.2 Interface to other Agents

As illustrated in Figure 5.5, the internal agent architecture is equipped with an interface to other agents. This interface depends on the asynchronous communication through message passing.

Therefore, it consists of communication handlers of all the interactions imposed by the communication protocols. These protocols are discussed in details in chapter 6. In addition to the communication handlers, the interface to other agents exploits the yellow pages' mechanism, provided by agent platforms. This yellow pages' mechanism allows agents to register themselves under a certain service and to be retrieved dynamically. Accordingly, the communication among the agents can be made more flexible. In this way, the fulfillment of the extensibility requirement, described in section 2.3, is significantly facilitated.

5.5.3 Control Subsystem

The introduction of this subsystem in the internal architecture caters for achieving the required balance between flexibility, in terms of reactivity, and efficiency. This balance is required from every agent involved in the optimization of schedules. i.e. all agent types except for material allocator agents and buffer agents. These two agent types are distinguished with their simple internal architecture, consisting of only the reactive layer which is directly connected with the environment. In other words, no control subsystem is necessary in case of material allocator and buffer agents.

In all other agent types, the control subsystem is directly connected with the interfaces to the environment and to the other agents. All incoming events are directed to the controller which decides on the layer to be activated. The decision on the activation of the available two layers depends mainly on the current scheduling mode. Under the schedule repair mode, the reactive layer is activated to ensure the reaction to the environmental dynamics in reasonable time. Under the schedule generation mode, the deliberative layer is activated for all agents that are involved in optimizing the schedule, as will be detailed in chapter 7.

5.5.4 Reactive Layer

This layer is responsible for exhibiting reactivity to the environmental dynamics. The concrete task of the layer differs from one level to the other of the multi-agent architecture. However, the objective of the layer remains the same, which is performing the required allocation. At the resource level, the allocation is conceived in terms of the allocation of the resource time to the jobs. This corresponds to solving the sequencing sub-problem for the machines and the routing sub-problem for the AGVs. Likewise, material allocator agents allocate material to jobs over time and buffer agents allocate store space to jobs over time. At the service level, the allocation corresponds to the selection of resources to be allocated to jobs. At the job and the job group levels, the allocation reacts to the received offers by modifying their local schedules. The self model captures the current status of the corresponding physical or logical entity in addition to its local schedule. The current status corresponds to the dynamic influencing factors captured by

the interface to the environment. For example, upon the detection of a machine failure, the self model is updated to reflect that the resource is broken down.

5.5.5 Deliberative Layer

The deliberative layer aims at fulfilling the efficiency requirement, discussed in section 2.3, based on an optimization subsystem. Its task is to optimize the schedule based on sophisticated optimization techniques. As illustrated in Figure 5.5, the optimization subsystem utilizes scheduling knowledge, encompassing formalizations of the knowledge necessary for the optimization. The optimization takes different forms along the different levels of the hierarchy. On the one hand, the aim of this layer is to generate near optimal schedule alternatives at the resource and the service levels. On the other hand, the optimization at the job and the job group layers aims at optimizing the selection of the available offers. The optimization process will be extensively discussed in chapter 7.

In this chapter, the architecture of the proposed agent-based scheduling was developed. An environmental model for representing the static details of FMSs, that are relevant to scheduling, was described. The different layers of the proposed architecture along with their associated agent types were derived. The internal agent architecture was explained and illustrated.

6 Inter-Agent Cooperation and Communication Protocols for Schedule Generation

It has been shown that according to the triggering event, a schedule is either generated or repaired. These two modes are handled in the rest of the thesis by addressing the schedule generation in the current and the following chapters and the schedule repair in chapter 8. In this chapter, the inter-agent cooperation and communication protocols for the schedule generation are described and illustrated. Section 6.1 derives the different cooperation levels among the agents based on an analysis of the previously presented influencing factors. Sections 6.2 and 6.3 explain the communication protocols for generating production and integrated schedules, under consideration of the identified cooperation levels.

6.1 Cooperation Levels among Agents

As previously mentioned, the schedule generation mode is classified into integrated scheduling and production scheduling, which is further classified into sequential scheduling and simultaneous scheduling. Every scheduling mode is triggered by a certain environmental configuration and hence the associated communication protocols and scheduling techniques are devised in a way that matches the corresponding configuration. The underlying principle of having different communication protocols for the different modes is to enhance the reactivity by allowing the concerned agents to participate in the decision making process, only when it is necessary. These different communication protocols are conceived at different abstraction levels of the proposed architecture, resulting in different cooperation levels along the hierarchy. As illustrated in Figure 6.1, three different cooperation levels are necessary for generating schedules under the different schedule generation modes and environmental conditions. At the lowest level of cooperation, the participation of agents at the resource and the service layers is sufficient to generate the schedule. At the next higher level, i.e. level 2, the participation of job agents is necessary for the schedule generation. At the highest cooperation level, i.e. level 3, concerned agents along the four layers of the architecture have to participate in the schedule generation.

Table 6.1 derives the required cooperation levels from the possible scheduling modes and the number of job groups associated with the requested jobs. This added factor is essential in determining whether the participation of job group agents is necessary or not, as will be detailed in this section. Under the sequential production scheduling mode, the scheduling request is initiated from a single job, which intuitively belongs to a single job group. This is handled by generating the schedule through the cooperation among the operation agents and the machine

agents at the service and the resource layers. Accordingly, cooperation level 1 is adequate for this case. The multiple job group case does not apply to the sequential production scheduling mode and is thus denoted as not applicable (NA). Under the simultaneous production scheduling mode, multiple jobs are available and can either belong to the same job group or to different job groups. In the former case, the participation of the concerned jobs is necessary to account for possible conflicting preferences among these jobs, which implies the suitability of cooperation level 2. In the latter case, level 2 is not sufficient due to the need to involve the concerned job group agents in the schedule generation to take the shared constraints into consideration. Consequently, cooperation level 3 is necessary for generating a production schedule for multiple jobs belonging to multiple job groups.

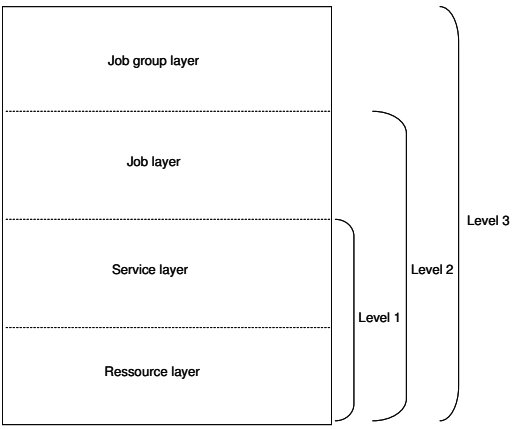


Figure 6.1: Agent-based cooperation levels with the associated abstraction levels of the agent-based architecture.

Table 6.1: Deriving cooperation levels from the scheduling mode and the number of job groups

	Number of job groups	
Scheduling mode	Single	Multiple
Production scheduling (sequential)	Level 1	NA
Production scheduling (simultaneous)	Level 2	Level 3
Integrated scheduling	Level 2	NA

Under the integrated scheduling mode, which is only supported for sequential scheduling, a production schedule has to be integrated with transportation schedule. For this purpose, the scheduling decision is escalated to the job level to allow the concerned job agent to evaluate the proposals from the operation and the transportation agents out of its own perspectives. In this

case, the decision spans the three lowest levels of the architecture and thus requires cooperation level 2.

6.2 Communication Protocols for Generating Production Schedules

In this section, the communication protocols required for generating a production schedule are illustrated. For every cooperation level, an interaction protocol based on the contract net (CNET) protocol is applied. This communication protocol is a high level protocol for distributed task allocation in agent-based systems, following the market-based contracting metaphor [Wool01]. It is widely applied due to its simplicity and its capability to achieve efficient cooperation among agents [Ouel03]. This protocol consists of three basic steps: task announcement, bidding and contracting. In task announcement, an agent that needs to achieve a certain task seeks help from other agents that are capable of accomplishing this task and contacts them. In bidding, every contacted agent responds to the task announcement by an offer, referred to as a bid, to solve this task. In contracting, the announcer of the task selects the agent with the most appropriate bid and awards it the responsibility of carrying out the task. It is also possible to contract with more than one agent at the same time, in case the parallel execution of the task at hand is desired [Wool01].

6.2.1 Communication Protocol for Cooperation Level 1

As previously mentioned, cooperation level 1 is associated with the sequential scheduling mode, where a single job request has to be satisfied. Similar to the conventional CNET protocol, this protocol involves task announcement, bidding and contracting steps, as illustrated in Figure 6.2. However, the task announcement is performed at two levels: by informing the operation agents and the machine agents. The optimization technique applied in case of a single job request depends on the assessment of the proposals gathered from the entire set of machine agents, corresponding to the currently available machines that provide the concerned operation. Accordingly, the task of incorporating the incoming job request into the schedule is announced to the entire set of machine agents. Based on the proposed bids and an allocation heuristic, which is explained in chapter 7, a subset of these machines is selected and this selection is confirmed by contacting both the machine agents and the job agent.

In spite of the participation of the job agent in the communication protocol, it in not involved in the decision making process underlying the schedule generation. Its participation is limited to announcing the task and receiving the confirmation. This explains the reason for conceiving this protocol under cooperation level 1, which involves the participation of agents at the resource and the service layers in the decision making process of the schedule generation. It is worth

noting that this limited participation of job agents is realized by the activation of the reactive layer in their internal architectures.

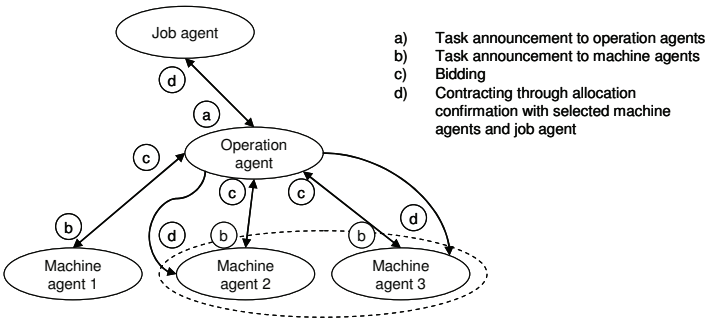


Figure 6.2: Communication protocol for production scheduling at cooperation level 1

6.2.2 Communication Protocol for Cooperation Level 2

To accommodate for the job-related evaluation, required for cooperation level 2, the CNET protocol is extended, as illustrated in Figure 6.3. The proposed protocol involves a two-level task announcement, a two-level bidding, an extra step for voting on the bids and a contracting step. Similar to the interaction protocol for cooperation level 1, an announcement of the scheduling request to the operation agents followed by an announcement to the machine agents is necessary. Nevertheless, the underlying optimization technique dictated by the simultaneous scheduling mode is different from the case of a single job. This is reflected in the interaction protocol by the task announcement to the machine agents. Whereas in case of a single job request all corresponding machine agents are contacted, in case of multiple job requests only a subset of the machine agents are requested to participate in the bidding mechanism. This is attributed to the fact that the evaluation of bids is preceded by the selection of the machines requested for bidding, in the applied optimization algorithm for the multiple job case, as will be detailed in chapter 7.

The main extension to the CNET protocol lies in the voting step, whose aim is to let job agents deliver their relative preferences of the different operation bids. This allows the operation agents to get an insight into the expected performance of the schedule to be selected based on the preferences of the job agents to the offered schedule alternatives. Based on the gathered votes of the offered bids, the operation agents select the final bid and confirm it with the concerned machine and job agents.

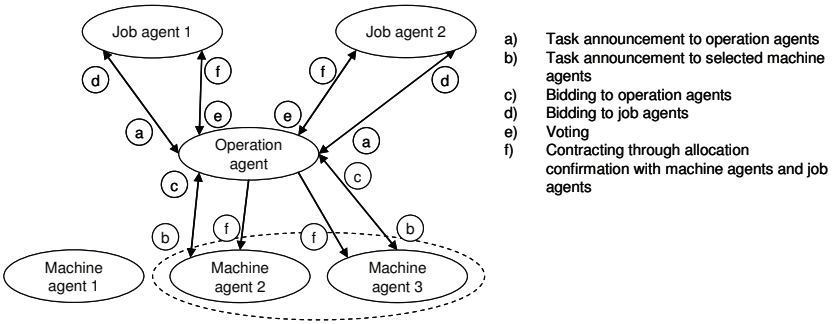


Figure 6.3: Communication protocol for production scheduling at cooperation level 2

6.2.3 Communication Protocol for Cooperation Level 3

For the accommodation of the job group perspective in the schedule evaluation, the proposed interaction protocol, described in the previous section, is further extended by adding two steps. As depicted in Figure 6.4, the added steps, namely step e and f, correspond to forwarding the bids gathered by the job agents to the job group agents and voting for these bids out of the job group agents' perspective.

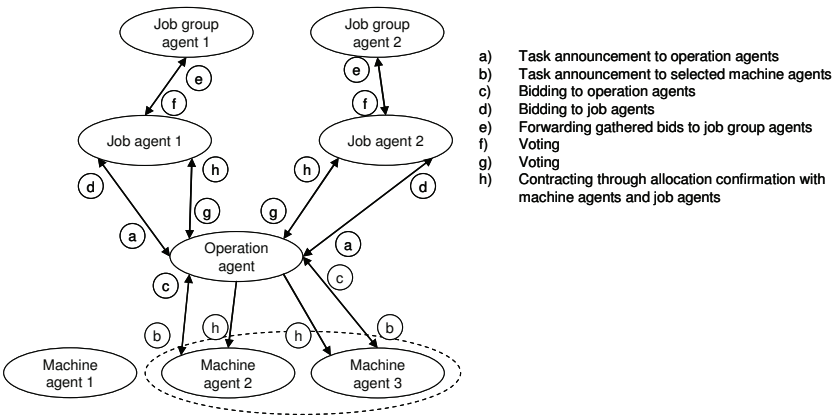
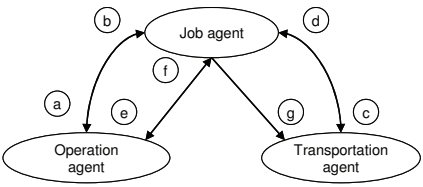


Figure 6.4: Communication protocol for production scheduling at cooperation level 3

6.3 Communication Protocol for Generating an Integrated Schedule

This section focuses on the generation of a schedule under the integrated scheduling mode. The generation of an integrated schedule takes place through the interaction between a job agent and the concerned operation and transportation agents. This interaction corresponds to cooperation level 2, as illustrated in Table 6.1. The proposed interaction protocol represents an extension to the CNET protocol and is depicted in Figure 6.5, by focusing on just the job and the service levels. As depicted in Figure 6.5, the integrated scheduling is triggered by a request sent from the job agent to the agents representing the required operations, according to the predefined plan of the associated part type. This step corresponds to the task announcement of the conventional CNET protocol. The contacted operation agents react by supplying the job agent with their bids in the form of production schedule alternatives. These bids are generated through the interaction with the machine agents at the lower level.



a) Task announcement to operation agents (production scheduling)
b) Bidding (production scheduling)
c) Task announcement to transportation agents
d) Informing initiator with transportation offers
e) Voting on production scheduling bids
f) Confirming production schedule
g) Contracting through confirming proposed transportation schedule

Figure 6.5: Communication protocol for generating an integrated schedule

Having received the production scheduling bids, the job agent pursues the process by contacting the transportation agents of the FMCs to which the proposing machines belong. Similar to production scheduling, transportation scheduling is performed through the interaction between the contacted transportation agents and other agents, representing the relevant transportation resources. For every production scheduling bid, a transportation offer is proposed, entailing the transportation costs, as will be explained in details in chapter 7. These transportation offers are used to evaluate the production scheduling bids from the job's perspective. The evaluation takes the form of votes corresponding to the relative preference of the offered production schedules. These votes are sent to the operation agents which had supplied the production scheduling bids to decide on the schedule to be selected. Eventually, the selected bids are confirmed with the requesting job agent, which in its turn confirms its selection of the associated transportation offers with the offering transportation agents.

The first part of the integrated scheduling, corresponding to the interactions among the operation and the machine agents to generate a production schedule, has already been explained in the

previous section. The interactions among the concerned agents in generating transportation offers are governed by the communication protocol shown in Figure 6.6

The protocol starts with a task announcement step initiated by the job agent to the transportation agents, offering the required transportation service. These agents are determined based on the physical layout of the machines associated with the production scheduling bids. For every FMC containing an offering machine, the transportation agent responsible for the transportation within this FMC is contacted.

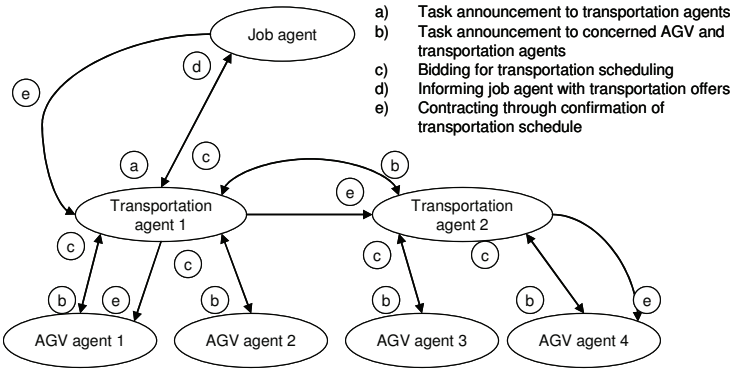


Figure 6.6: Communication protocol for transportation scheduling

Upon receiving a job request, a transportation agent gathers bids from AGV agents to transport the workpieces to the required destination, in case it is located in the current FMC, or to a neighboring cell otherwise. In the latter case, the transportation agent of the neighboring cell is requested to provide an offer for transporting the workpieces either to the destination, if it is in its cell, or to other neighboring cells along the route to the destination. This process continues until the destination is found and the proposal propagates back to the transportation agent initiating the request, which calculates the path and its associated cost. In calculating the required route, AGV agents communicate with buffer agents and material allocator agents, under consideration of the physical layout. This communication takes the form of the traditional CNET protocol, as will be illustrated in this section. The best found path along with the associated cost is sent as an offer to the requesting job agent, which in its turn uses this offer in evaluating the production scheduling bids. Upon receiving a confirmation of the selected production schedule from the concerned operation agents, the job agent confirms the associated transportation proposal with the offering transportation agent, which in its turn confirms the proposal with the associated transportation and AGV agents.

To illustrate, consider the example of Figure 6.7. Assuming that a job agent seeks a transportation bid from machine M1 to machine M6, it has to send a request to T1, i.e. the transportation agent in charge of the FMC containing the source machine M1. The communication among the agents in generating a transportation schedule is depicted in Figure 6.8, where the arrows are labeled with the step number of the communication protocol. As illustrated in Figure 6.8, the task announcement is initiated by the job agent and propagates to the agents, representing the material handling resources in FMC2, where the destination machine M6 is located.

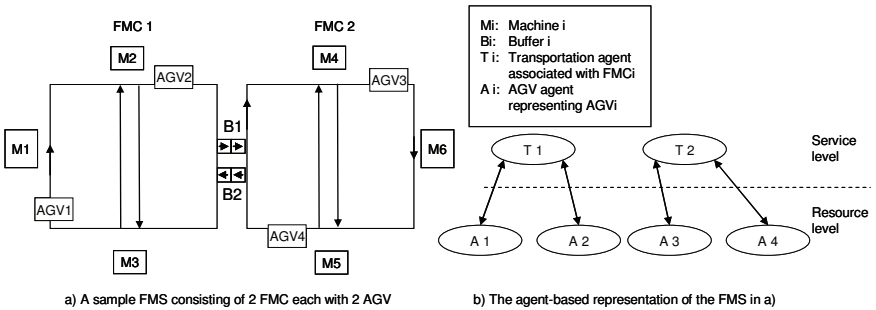


Figure 6.7: The agent representation of the transportation resources of a sample FMS

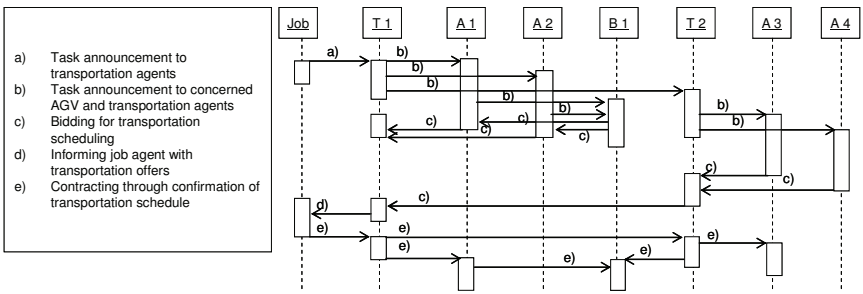


Figure 6.8: A sequence diagram illustrating the communication protocol for generating a transportation schedule

The task announcement takes the form of a transportation job with predefined source and destination. In specifying the source and destination, a transportation agent takes the layout of the shop floor into consideration. A job for transporting workpieces from a source machine to a destination machine may be broken up into sub-tasks over a set of cells. In each of the cells where neither the source nor the destination is located, a transportation task corresponds to moving the workpieces from an input buffer to an output buffer inside the same cell. In the considered example, the job of transporting workpieces from M1 to M6 is decomposed into two

sub-tasks: transporting workpieces from M1 to B1 and from B1 to M6. While the former task is undertaken by T1 the latter is assigned to T2.

Each AGV agent reacts to the task announcement by supplying a proposal, comprising the collecting time of the workpieces from the source of the associated sub-task and the delivery time at the destination of the same sub-task. As illustrated in Figure 6.8, for A1 and A2 agents to prepare their proposals, they have to contact B1 to get the earliest time at which the required store space would be available. Eventually, AGV bids are collected by the concerned transportation agent and a transportation proposal is prepared and supplied to the requesting job agent. It is then up to the job agent to accept or refuse the proposal. Refusing the proposal implies selecting different source and/ or destination machines. In case the proposal is accepted, the job agent confirms with the offering transportation agent which in its turn confirms with the concerned AGV and transportation agents, as illustrated in Figure 6.8. This example has considered the partial transportation from a machine to another machine. When considering the complete transportation of the workpieces of a given job, an offer is requested for every transportation step starting from the material store all the way up to the final product store.

This chapter has addressed the interactions among the agents in generating schedules. The effect of the different schedule generation modes and their associated environmental conditions were analyzed. This resulted in developing a model, describing the different cooperation levels involved in the schedule generation. Each cooperation level corresponds to the participation of agents, at certain layers of the proposed agent-based architecture, in the decision making of the schedule generation. In light of this model, the communication protocols of the different scheduling modes along the different cooperation levels were explained and illustrated.

7 Hierarchical Agent-based Schedule Generation

This chapter embarks on the details of the schedule generation process. A hierarchical agent-based schedule generation process is proposed. The chapter is organized as follows. Section 7.1 gives an overview on the proposed schedule generation process by highlighting the decomposition of the schedule generation into sub-tasks and assigning these sub-tasks to the different layers of the agent-based architecture. These sub-tasks are classified into two main categories, namely the search for schedule alternatives and the evaluation of these alternatives. Section 7.2 is devoted to explaining the search for production schedule alternatives, which constitutes the core of the schedule generation process. Section 7.3 addresses the search for integrated schedules. The evaluation and selection of schedule alternatives are explained in sections 7.4 and 7.5.

7.1 Overview on the Proposed Schedule Generation

The basic idea of the agent-based schedule generation is to tackle the problem complexity by decomposing it into sub-problems and assigning these sub-problems to the concerned agents. Each agent endeavors to solve its assigned task and reach its local objective. Agents depend on their local knowledge of the problem and can thus lead to conflicts or reduced performance from the global view. To avoid potential conflicts and converge to a near-optimal schedule, concerned agents have to work together at different levels of cooperation, as explained in the previous chapter.

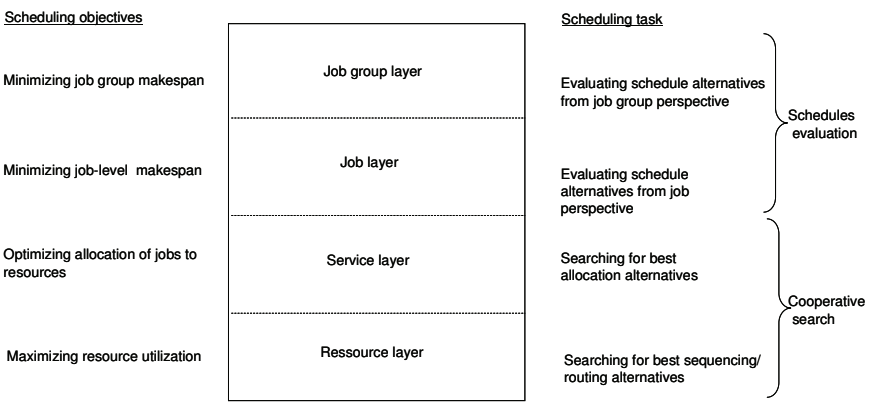


Figure 7.1: The proposed schedule generation process

The proposed schedule generation process represents a bottom up process starting with the resource agents going up all the way to the job group agents. At each level, a given objective is endeavored and a certain sub-task is to be achieved, as illustrated in Figure 7.1. At the resource level, the optimization objective is to maximize the utilization of the concerned resources. This is pursued through solving the sequencing and the routing sub-problems, which represent sub-problems of production scheduling and transportation scheduling respectively. Due to their intermediary role between job agents and resource agents, agents at the service level aim at accounting for the objectives of both the jobs and the resources while optimizing the allocation of resources to jobs. By solving the two search tasks at the two levels, schedule alternatives can be generated under consideration of the given objectives. The search at the two levels is an iterative process and is referred to as the cooperative search.

The evaluation at the job level aims at influencing the schedule selection, carried out at the service level, in a way that yields a schedule that minimizes the makespan of the individual jobs. By allowing every involved job agent to evaluate the schedule alternatives out of its own perspective, the selected schedule can result in the best makespan for the majority of jobs and hence to a good overall makespan. Following the same principle, the schedule generation process accounts for the evaluation of the involved job group agents, whenever cooperation level 3 is necessary. This evaluation aims at taking the makespan of the involved job groups under consideration, when optimizing the schedule.

In the following sections, a more detailed discussion about the cooperative search and the evaluation phases of the schedule generation is given.

7.2 Cooperative Search for Production Scheduling Proposals

In pursuing their scheduling objectives, machine and operation agents perform the cooperative search process, depicted in Figure 7.2 by distinguishing its sub-functionalities with dotted rectangles. According to the desired cooperation level, the results of this search is either used by the operation agent to select the best schedule alternative found so far or delivered to the concerned jobs, at the layer above, for evaluation. While the former case corresponds to the behavior of agents under cooperation level 1, the latter case is valid for cooperation levels 2 and 3.

In what follows, the cooperative search for both the sequential production scheduling and the simultaneous production scheduling is explained. The two other functionalities, shown in Figure 7.2, are handled separately in section 7.5.

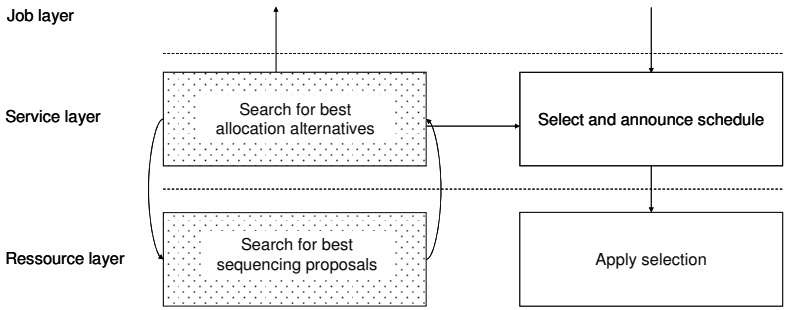


Figure 7.2: Schedule generation functionalities at the resource and service levels

7.2.1 Searching under the Sequential Production Scheduling Mode

The search for production schedule alternatives under the sequential scheduling mode reduces to the trivial case of allocating the required resources to a single job. As illustrated in Figure 7.3, the process is triggered when an operation agent receives a single job request. It reacts by requesting proposals from the corresponding machine agents. Every contacted machine agent takes this request and offers an allocation proposal on this machine. The simple dispatching rule first come first served (FCFS) is applied to avoid introducing disturbances to the already scheduled jobs. The offered proposal comprises the earliest start time of the job on the concerned machine and the associated allocation cost. For the sake of minimizing the time lost in setup, the allocation cost is defined to denote the time lost by a machine in switching to a new part. This cost is associated with an allocation request of an incoming part on a certain machine and amounts to the setup time that was incurred by the machine for the previous allocation and would be lost if the machine is to process the new request. In this way, the impact of potential fluctuations in the part types of the incoming job requests is minimized.

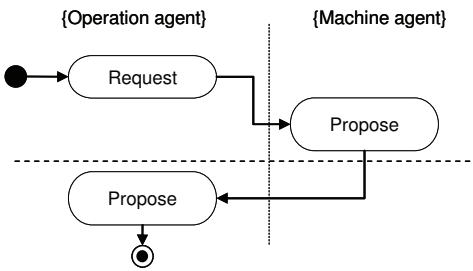


Figure 7.3: An activity diagram illustrating the cooperative search process under sequential production scheduling mode

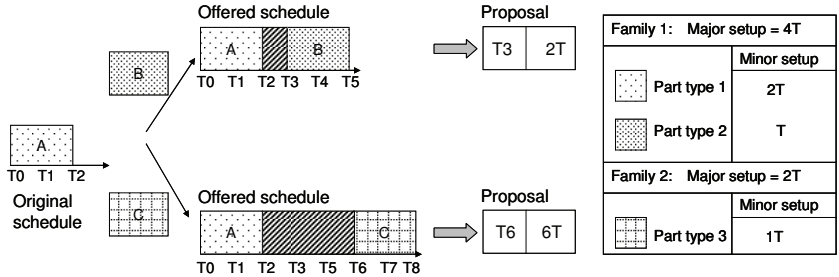


Figure 7.4: A scenario illustrating the preparation of a machine proposal

To illustrate, consider the scenario depicted in Figure 7.4 for preparing proposals on a certain machine for two different allocation requests. In the first case, the job to be scheduled, job B, belongs to the same part family of the last job scheduled on this machine. Incorporating this job into the schedule implies losing the minor setup which was incurred for part type1 and incurring minor setup for part type2, to which the incoming job belongs. Accordingly, the offered allocation would be associated with a cost amounting to the lost setup which is 2T, corresponding to the minor setup of part type1. This cost together with the start time offered to job B, which is in this example T3, constitutes the proposal offered for job B. For the other request coming from job C, the cost would be higher since C belongs to a different part family and hence necessitates losing the setup that was incurred for part type1 of part family1. The lost setup in this case amounts to six time units corresponding to the major setup time of part family1 and the minor setup time of part type1.

7.2.2 Searching under the Simultaneous Scheduling Mode

The cooperative search process under the simultaneous production scheduling mode tackles the problem complexity by employing genetic algorithms both at the resource and the service levels. As illustrated in Figure 7.5, the process consists of four steps: initialization, optimization, validation and termination [BaGö10].

7.2.2.1 Initialization

The initialization is initiated by the operation agents, upon receiving job requests. Operation agents undertake the initialization of the cooperative search and trigger their machine agents to perform an initialization for their local search.

Initializing Cooperative Search

The objective of this initialization is to derive the number of machines to be considered for the allocation and the number of job lots to be allocated to each machine. This is calculated as

follows. First, the total number of job lots K is calculated by multiplying the number of jobs J by a predefined parallelization rate r, corresponding to the number of machines to be allocated to each job. For instance, if the number of jobs amounts to 20 and the parallelization rate r is set to 2 the total number of job lots would be 40. The parallelization rate in this case corresponds to the simultaneous allocation of every job to two machines for the parallel execution of the operation in concern.

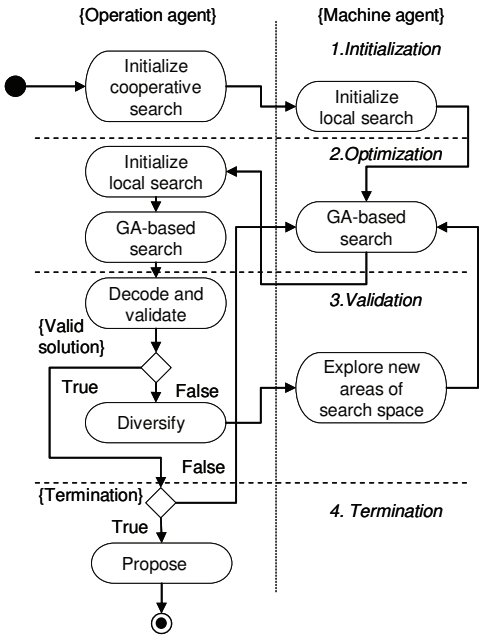


Figure 7.5: An activity diagram illustrating the cooperative GA-based search process under the simultaneous production scheduling

Second, the number of machines n, to be considered for allocation, and the associated chromosome length l, corresponding to the number of jobs allocated for every machine, are derived by the iterative procedure, listed in Figure 7.6. Initially, the total number of machines N is considered for allocation. The chromosome length l is calculated by rounding up the result of dividing K by N. In case the resulting number of jobs is below a predefined minimum length L, the number of machines is decremented by 1 and l is calculated again. This procedure iterates until l reaches L or the number of machines reaches 1.

```
K = J * r
n= N
l= ceil (K/n)
While (l < L and  n>1)
begin
     n= n-1
     l= ceil (K/n)
end
```

Figure 7.6: Initialization algorithm for the cooperative search

To illustrate, consider the previous example and assume the availability of 20 machines. Allocating the 40 job lots to the 20 available machines would result in every machine having 2 lots. This implies 2 jobs at each machine which is relatively small and can lead to a changeover of the current setup of the machines to fulfill the current job requests. In this example, if we suppose a minimum length of 4, only 10 machines would suffice to reach the required parallelization rate. However, higher machine utilization can be attained by configuring the parallelization rate. In the case of multiple job requests, it is assumed that all machines are initially unloaded, i.e. no jobs are scheduled on the machines. Accordingly, it would not make sense to favor one machine to the other and hence the contacted machines are selected at random, from the machines performing a certain operation.

Initializing Local Search

Machine agents receive from their operation agents the number of jobs to be scheduled along with their identifiers. Every machine agent selects l jobs out of the total J jobs at random and generates a population of chromosomes encoding possible permutations of the selected jobs. For example, the chromosome {2,7} encodes a local schedule for executing job 2 followed by job 7.

Having generated the chromosomes, the fitness of each of these chromosomes is calculated. As a measure of the relative fulfillment to the optimization objective of the chromosomes, the fitness function at the resource level aims at assessing the machine utilization implied by the chromosome. Since different chromosomes represent different permutations of the same set of jobs, the total processing time for all chromosomes is the same and is thus disregarded from the fitness calculation. Accordingly, the fitness function $f_m(c_m)$ of a given machine chromosome c_m is calculated by adding up the setup required for every job in the sequence of jobs encoded by this chromosome, as in (7.1).

$$f_m(c_m) = \sum_j s(j) \tag{7.1}$$

Different sequences of the same set of jobs are associated with different setup times. This depends on whether succeeding jobs refer to the same part type and part family or not. The objective at the machine level is to minimize the function calculated in (7.1).

7.2.2.2 Optimization

As depicted in Figure 7.5, the optimization step consists of the GA-based search at the resource level followed by the initialization and the GA-based search steps at the service level.

GA-based Search at Resource Level

Every machine agent searches for the best sequence of l jobs. The search for an optimal sequence is carried out by applying standard GA operators to the current population to evolve better offspring. The quality of the offspring is measured by its fitness function. As a minimization function, the lower the fitness function is, the better the quality of the chromosome is deemed to be. The evolution is pursued until the termination condition is reached, which is defined in terms of a given number of generations. Having reached the termination condition, each machine agent delivers the best c chromosomes to the operation agent along with the average fitness of the current population.

Initializing Local Search at Service Level

This step is iteratively performed during the cooperative search whenever the operation agent receives solution alternatives from the contacted machine agents. Therefore, this step is regarded as part of the search step and not as part of the overall initialization step which is performed only once. During the initialization step, combinations of offers from the resource agents are generated.

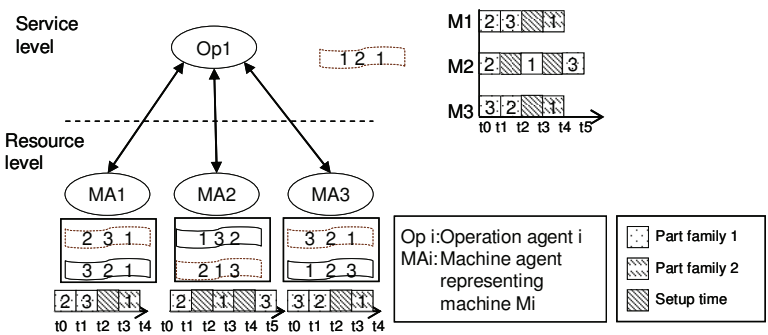


Figure 7.7: Encoding of chromosomes at the resource and service levels

The encoding of chromosomes at the service level takes the form of indices to the offered machine chromosomes. As illustrated in Figure 7.7[4], a chromosome at the service level with the value {1,2,1} encodes the schedule representing the combination of the first chromosome from

[4] The dashed boarders distinguish chromosomes, whose decoded schedules are shown.

MA1, the second chromosome from MA2 and the first chromosome from MA3, in the c offered chromosomes. This encoding ensures a 1-1 mapping of chromosomes to valid schedules.

Similar to the initialization step at the resource level, chromosomes are evaluated based on a fitness function at the service level. The aim of this fitness function is to optimize the allocation of machines to jobs by accounting for the optimization objectives both at the resource and the job levels. Therefore, the fitness of operation chromosomes is calculated as a weighted sum of two terms, as in (7.2). The first term corresponds to the job-related criteria and measures the fairness of job allocation on the machines. This fairness is estimated based on the standard deviation of the lot sizes L_o of all jobs allocated on the machines for a given operation. As in (7.3), the lot size l_{j_o} of a given job j on a given operation o is calculated by simply dividing the quantity of the job by the number of machines allocated to this job according to the chromosome in concern. The second term is an estimate of the machine utilization which is again measured in terms of the setup time. However, at the service level, the maximum setup time incurred by all the machines is regarded as an indicator of the utilization of the allocated machines. The objective of an operation agent is to minimize $f_o(C_o)$, calculated in (7.2).

$$f_o(c_o) = w \cdot stdv(L_o) + (1-w) \cdot Max(F_m(c_o)) \tag{7.2}$$

$$L_o = \{l_{j_o} \mid j \in J, l_{j_o} = \frac{Q(j)}{n_{jo}}\} \tag{7.3}$$

Where:

w	Predefined weight
stdv(x)	Standard deviation of set x
L_o	Set of lot sizes on operation o
$F_m(C_o)$	Set of fitness values of the machine chromosomes constituting the operation chromosome C_o
Q(j)	Quantity of job j
n_{jo}	Number of machines allocated for job j on operation o

GA-based Search at Service Level

The search for the best combination of machine proposals at the service level is confronted with c^n solution candidates constituting the solution space. Since this number grows exponentially by increasing the number of the selected solutions, c, or the number of machines considered for allocation, n, an exhaustive exploration of the entire set of solutions is only feasible in case of relatively small numbers of selected machines or selected solutions from each machine. Therefore, GA-based search is applied by evolving new generations from the current population.

7.2.2.3 Validation

Schedule alternatives combined by operation agents are the result of the local optimization at the individual machines. Therefore, the generation of a feasible solution from the operation

perspective is not guaranteed. That is why the schedule alternatives generated at the service level have to be validated based on the following steps.

Decode and Validate

The validity check applied at this stage concerns the inclusion of every job by at least one machine. If during the search, no valid chromosome could be found, the operation agent works on diversifying the populations of a subset of the machine agents.

Diversify

In case no chromosome represents a valid schedule, a number of machine agents are requested to diversify their populations. Machines requested to pursue their exploration of the search space are those having the worst average fitness. The percentage of the selected machines is directly proportional to the percentage of the missing jobs.

Explore New Areas of the Search Space

In this step, the local search space is diversified to explore non-explored areas of it, in an attempt to find a valid solution from the operation perspective. Diversifying the population of the selected machine agents takes place by increasing the mutation rate with a predefined diversification factor and pursuing the execution of the local evolution process. This adaptation of the mutation rate results in the investigation of unexplored areas of the solution space and regaining missing jobs. Having reached the termination condition, these machine agents deliver their best chromosomes to their operation agent. Operation agents pursue their search for valid chromosomes as before.

7.2.2.4 Termination

Having generated valid solution candidates at the service level, operation agents check the termination condition. The termination condition can depend on the quality of the current generation. This is measured by comparing the quality of the best or the average fitness of the current population with the one from the previous valid population. In case no improvement is achieved, the process can terminate by considering the previous population. However, this may lead to a local optimum. Therefore, a predefined number of iterations is considered by the termination condition to improve the probability of reaching a near-optimal solution.

7.3 Generating Integrated Schedules

In this section, the generation of integrated schedules is addressed by focusing on the generation of transportation schedules and how they are integrated with production schedules.

7.3.1 Steps of Integrated Schedules Generation

As previously explained in chapter 6, the schedule generation process under integrated scheduling is governed by the concerned job agents, which are in charge of integrating production schedules with transportation schedules. This integration is governed by the communication protocol, explained in section 6.3. In this section, the internal behavior of job agents, associated with this protocol, is described. This behavior can be summarized in the following steps [BSG10].

7.3.1.1 Gathering Operation Bids

According to the work plan associated with this job, calls for proposals (cfp) are sent to the corresponding operation agents. Each operation agent gathers, in its turn, offers from machine agents corresponding to the machines currently supporting the operation in concern. Evaluating machine offers is based on the offered start times along with the associated costs. Offers received by the job agents from the operation agents comprise the machine identifiers, the earliest start times, and the associated costs.

7.3.1.2 Gathering Transportation Offers

In generating an integrated schedule, job agents request a transportation offer for every two operations I, J in the work plan of the jobs, where I precedes J. To retrieve a transportation agent, the machine identifiers offered by operation I are first retrieved and their flexible manufacturing cells are identified. The transportation agent corresponding to each of these cells is then called for transporting workpieces from the machines specified in the operation offer of I to the individual machines proposed by the operation J. The same applies for transporting the workpieces from the load station to the machines of the first operation and transporting the workpieces from the last operation to the unload station.

Proposals offered by transportation agents include the collecting time from the source and the delivery time at the destination along with the associated transportation costs. The transportation costs correspond to the additional time spent in transporting the requested job, under consideration of the speed and capacity of the AGV at hand, as will be detailed later in this chapter.

7.3.1.3 Evaluating Offers

Offers retrieved from operation and transportation agents are compiled together to generate alternative integrated schedules for the concerned job. For every production schedule alternative, the associated transportation costs are considered to calculate the overall costs of the integrated schedule. Production schedules are sorted based on their makespan and their associated overall costs. In this way, offers that minimize the makespan and maximize the

utilization of the involved resources are favored during the selection. This step is described in more details in section 7.4.

7.3.1.4 Reporting Preferences to Operation Agents

Having sorted the offers in the previous step, the job agent delivers this sorted list to the corresponding operation agents. Each operation agent confirms the allocation with the machines, whose offers are associated with the highest preferences. The determination of the number of machines to be selected for allocation is made possible by applying heuristics, which are explained in 7.4.

7.3.1.5 Confirming Allocation with Transportation Agents

Upon receiving a confirmation of allocation from the operation agents, the job agent confirms with the concerned transportation agents, which in their turn confirm with the associated AGV and transportation agents. In case the production schedules of the consecutive machines of the integrated schedule do not allow for the associated transportation time, the schedules of the machines, that conflict with the transportation schedule, are corrected. This correction takes the form of delaying the allocated slot of the job at the concerned machine, to fit with the associated transportation schedule.

7.3.2 Transportation Scheduling

In this section, the step corresponding to gathering the transportation offers is elaborated on to conceive how a transportation schedule is generated. A transportation schedule is generated through the interactions among AGV agents, transportation agents and job agents. Upon receiving a request for gathering transportation bids, a transportation agent gathers bids from AGV agents to transport the workpieces to the required destination, in case it is located in the current FMC, or to a neighboring cell otherwise. In the latter case, the transportation agent of the neighboring cell is requested to provide an offer for transporting the workpieces either to the destination, if it is in its cell, or to other neighboring cells along the route to the destination. This process continues until the destination is found and the proposal propagates back to the transportation agent initiating the request, which calculates the path and its associated cost.

Table 7.1: An example of an internal schedule of an AGV agent

Job	Source	Target	Collecting time	Delivery time	Quantity
1	M3	M1	0	5	3
2	M2	Buffer2	3	7	2
3	Buffer1	M1	6	11	1

Every AGV agent keeps a list of scheduled jobs with their associated collecting and delivery times and quantities, calculated in terms of the number of palettes, as illustrated in Table 7.1. When an AGV agent receives a request for transporting a new job, it goes through its internal list searching for the earliest possible time it can offer under consideration of its own capacity. The A* algorithm is applied for calculating the shortest path between the predefined source and destination. The transportation cost is estimated based on the AGV speed and the additional paths that the AGV has to take in satisfying a transportation request.

The transportation agent initiating the cfp gathers the proposals from its AGV agents and from the contacted transportation agents, as illustrated in the previous chapter. These proposals are compiled to end up with a complete proposal for satisfying a transportation job with a predefined source and destination. A proposal is represented in terms of a collecting time from the source and a delivery time to the destination along with an associated cost C2 which is calculated by summing up all the proposal costs C1 of all AGV agents involved in the final proposal. A proposal is confirmed with all the concerned agents, once the transportation agent initiating the cfp receives a confirmation from the job agent.

7.4 Evaluating Schedule Alternatives

Having discussed the generation of both production and integration schedules, this section focuses on the evaluation of the generated schedule alternatives. As captured in Figure 7.1, the evaluation of schedule alternatives takes place at two different levels of abstraction, namely the job level and the job group level. The functionalities performed at these levels are schematically presented in Figure 7.8. As illustrated in Figure 7.8, proposals gathered from agents at the service layer are combined by the job agents. Afterwards, they are either directly evaluated and sent back to the concerned operation agents along with their evaluation values or raised to the job group layer for evaluation from the job groups' perspective. This depends on whether the current cooperation level is level 2 or level 3 respectively.

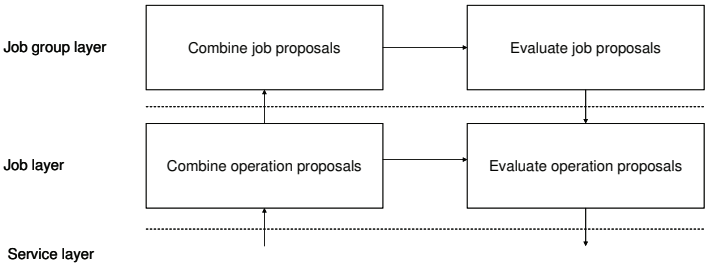


Figure 7.8: Functionalities of the evaluation at the job and job group layers

In what follows, the evaluation at the two layers is explained in more details.

7.4.1 Evaluation at the Job Layer

As depicted in Figure 7.8, operation offers are evaluated at the job layer by the concerned job agents by first combining these offers together and then evaluating them. As previously explained, this evaluation is necessary under the simultaneous production scheduling mode and the integrated scheduling mode.

7.4.1.1 Combining Operation Proposals

Under the simultaneous scheduling mode, operation proposals are offered to the concerned job agents in the form of start and end times. These offers are combined together to form schedule alternatives out of the job perspective. Figure 7.9 illustrates the propagation of production scheduling proposals from the resource level up to the job level. For illustration purposes, only one proposal from the considered agents is depicted in the figure.

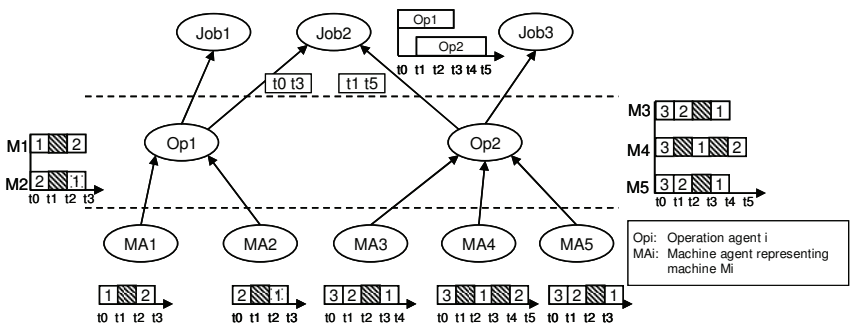


Figure 7.9: Combining production scheduling offers at the job level

Upon receiving the proposals from their resource agents, operation agents select the c proposals, with the best fitness values, and interpret them into job offers. This interpretation involves extracting the earliest start time and the latest end time for every concerned job. In the example depicted in Figure 7.9, job2 receives the offer {t0, t3} from op1, corresponding to the earliest start time and the latest end time of job2 on operation1, according to the considered schedule alternative. Similarly, the offer {t1, t5} is received from op2. The resulting schedule of job2 is depicted in Figure 7.9, yielding a makespan of five time units, corresponding to the difference between the latest end time and the earliest start time, under consideration of all the involved operations.

Under the integrated scheduling mode, operation offers are delivered to jobs by including information required for the generation of transportation schedules. For every operation offer, the offering machines and the associated costs are included. In the example of Figure 7.9, under the integrated scheduling mode, job2 would receive from op1 {{M1, t2, t3, c1}, {M2, t0, t1, c2}. This detailed offer encompasses the machine proposals in terms of their identifiers, their offered start times, their offered end times and the associated costs, which are calculated as explained in section 7.2.1. This detailed information is necessary for the job agent to calculate the transportation costs of the combined operation offers. For every possible combination of machine offers belonging to two consecutive operations, the transportation costs are collected by contacting the concerned transportation agents. Considering the example in Figure 7.9, combinations of offers include (M1, M3), (M1, M4) and (M2, M3). The combination (M1, M3) corresponds to processing the first operation at M1, and the second operation at machine M3.

7.4.1.2 Evaluating Operation Proposals

The evaluation starts with calculating the evaluation function for the combined operation schedule alternatives. While the evaluation function under the production scheduling mode is the makespan, the evaluation function under the integration scheduling mode is twofold consisting of the makespan and the overall costs. The costs of an integrated schedule account for both the machine and the transportation costs based on a weighted sum. The predefined weight W_T corresponds to the significance of transportation costs T_C relative to machine costs M_C. Accordingly, the costs of an integrated schedule φ is calculated as in (7.4).

$$\varphi = w_T \cdot T_C + (1 - w_T) \cdot M_C \tag{7.4}$$

The evaluated schedule alternatives are sorted based on the resulting values of the evaluation function. Finally, the sorted schedules are traversed, by starting from the schedule yielding the best value of the evaluation function. Every occurrence of an operation offer is rewarded by increasing its vote, which is initialized to zero. Eventually, these votes reflect the relative preferences of a job agent to the operation offers.

7.4.2 Evaluation at the Job Group Layer

The evaluation at this layer is performed similar to the evaluation at the job layer but out of the job groups' perspective. Since the optimization objective at the job group layer is to minimize the makespan of the associated group of jobs, the same criterion, i.e. the makespan of the job group, is adopted as the evaluation function. As previously explained, operation offers combined by job agents are either evaluated within the same job agent or raised to the job group layer to be combined and evaluated, depending on the value of the current cooperation level. This decision to limit the evaluation to the job level or to extend it to span the job group level, in addition to the job level, is governed by the protocol in Figure 7.10. This protocol clarifies that the evaluation is escalated to the job groups only under cooperation level 3.

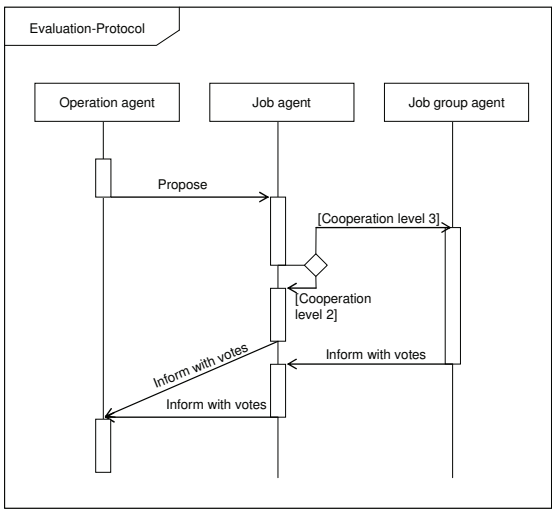


Figure 7.10: Proposals evaluation protocol

As illustrated in Figure 7.10, the evaluation at the job group level is performed first to the job proposals out of the job groups' perspective. Similar to the job-based evaluation, offers received from the lower layer, i.e. the job layer, are first combined, as illustrated in Figure 7.11. The combined offers are evaluated based on their resulting makespans. Accordingly, the offers collected from the jobs are then voted for, as in the job-based evaluation. The evaluation results of the job group agents are then propagated to the job agents, which in their turn update these results with the evaluation out of their own perspectives. Eventually, the final evaluation results are delivered to the operation agents for selecting the schedule having the highest overall evaluation, as will be explained in the next section.

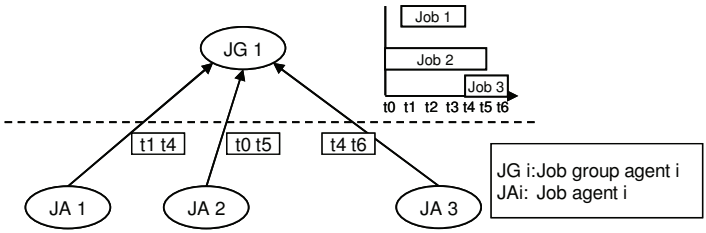


Figure 7.11: Merging offers at job group level

7.5 Selecting and Updating Schedules

The selection of the final schedule takes place at the service level by the concerned operation agents. It is worth mentioning that even under the integrated scheduling mode, schedules are selected by operation agents based on the evaluation of jobs which takes the transportation costs into consideration.

7.5.1 Schedule Selection

The selection of a schedule out of the schedule alternatives, found at the cooperative search step, differs according to whether the current scheduling mode is sequential or simultaneous.

7.5.1.1 Schedule Selection under Simultaneous Scheduling Mode

Under the simultaneous scheduling mode, machines are assumed to be initially unloaded, i.e. not allocated to any job. Therefore, the current setup of the machine does not influence the allocation decision and machines considered for allocation are selected randomly. Offers proposed by the selected machine agents are evaluated by the operation agents. For every schedule alternative, the votes from the different jobs are added up. The schedule alternative having the highest vote is selected. This schedule alternative is deemed to result in the best satisfaction of the evaluation criteria along the four levels of the hierarchy. In this way, the schedule is optimized from a global perspective.

7.5.1.2 Schedule Selection under Sequential Scheduling Mode

Under the sequential scheduling mode, machines are loaded, i.e. they have different schedules and are set up to different part types. Therefore, the machines to be considered for allocation are selected based on their proposals. This justifies the consideration of all the machine agents offering a certain operation, when sending a call for proposal. Based on these proposals, the machines to be allocated are selected. The number of machines to be considered for the allocation is specified based on the following heuristics.

Heuristic 1: Balancing Resource Distribution

This heuristic attempts to minimize the setup time by balancing the utilization of available machines on the different part families. This occurs by allocating a job from a certain part family to a number of machines proportional to the expected quotient of the amount to be produced from this family to the total number of parts along all planned families. Such information can be acquired and updated from planning. The machines to be selected are those offering the best proposals with the least allocation costs.

Heuristic 2: Greedy Allocation

Distributing parts to be manufactured among all machines selected by the previous heuristic can be inefficient for jobs with small volumes. This is due to the loss of long setup time for the sake of shorter processing time. To avoid this, the loss represented in the allocation cost is compared to the expected gain represented in the batch processing time. Accordingly, an allocation offer is accepted only if the expected gain exceeds the allocation cost. In the extreme case when the job's quantity is too small to yield a gain even on a single machine, the operation agent allocates the job to a single machine, namely the machine associated with the best proposal.

7.5.2 Updating Schedule

Updating the schedule is carried out by sending a command to the involved machine agents informing them with the index of the selected proposal. Similarly, the job agents are informed with their start and end times at the operation in concern. Job agents inform in their turn their job group agents with the selected schedule and the transportation agents in case of the integrated scheduling mode. In this way, the schedule is saved locally from the different perspectives. This decentralization of schedule helps enhancing the reactivity while reacting to the environmental dynamics, as shown in chapter 8.

This chapter has explained the hierarchical schedule generation process by explaining the objective and the tasks pursued at the different layers of the architecture. These decomposed tasks have been detailed by emphasizing the scheduling techniques applied at the different layers under the different schedule generation modes.

8 Conception of Disturbance Handling

Having covered the schedule generation mode in the two previous chapters, the focus is laid in this chapter on schedule repair by working out a concept for disturbance handling. Section 8.1 develops a model for manufacturing disturbances. The objective of this model is to analyze disturbances and to introduce for the conception of disturbance handling. Based on this disturbance model, a disturbance handling method is proposed in section 8.2. Details of extending the proposed agent-based approach with the disturbance handling method are explained in section 8.3. To illustrate the applicability of the disturbance handling method, application scenarios are demonstrated by considering the most frequently occurring disturbances in section 8.4.

8.1 Modeling Disturbances

A schedule embodies allocation dependencies between jobs and resources. An event causing or necessitating a change in a planned allocation can disturb the whole schedule and is thus called a disturbance. Examples of such events include the arrival of a rush order, a delay in the arrival of required material and a failure in one of the resources. Repairing a schedule requires a sophisticated analysis of the problem to identify the best possible solution within time limits. For this purpose, a disturbance model is proposed to aid in analyzing the different disturbances [BaGö09a].

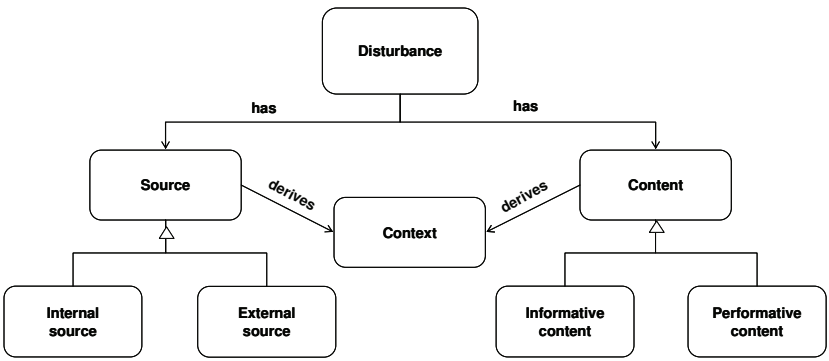


Figure 8.1: A disturbance model

As illustrated in Figure 8.1, a disturbance is modeled based on its source and content. The source corresponds to the entity which is causing or being directly affected by the disturbance. A source can be regarded as either internal or external depending on whether it is internally

controllable by the manufacturing system or exogenous to it. Internal disturbances stem from the shop floor control due to events like machine breakdowns or the underestimation of the execution time. On the other hand, external disturbances arise from changes in the production environment. These changes correspond to events such as the delay of the material supply.

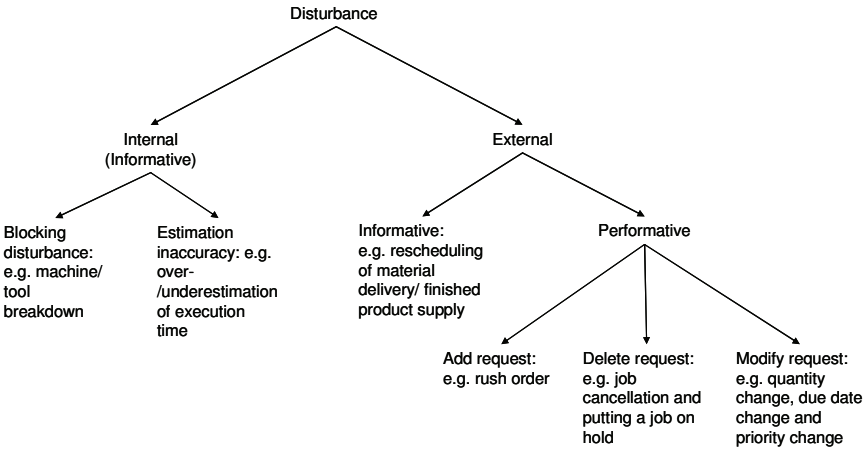


Figure 8.2: Taxonomy of disturbances

The content of a disturbance models the attributes directly describing its semantics. To illustrate, the content of a delay in material delivery would comprise an identifier for the delay event, an identifier of the delayed material in addition to the old planned date and possibly the new estimated date. Two types of content can be identified: informative content and performative content. While the former corresponds to reporting the occurrence of an event, the latter refers to a request or a desire for a certain modification to the schedule. This implies that while the deviation from the planned schedule occurs as a result of the event in the former case, it is associated with the action taken to handle the request in the latter case.

Figure 8.2 illustrates taxonomy of disturbances based on a classification according to the source followed by a further classification according to the content. As illustrated in the taxonomy, external disturbances are always coupled with informative content which can either correspond to a blocking disturbance like a machine breakdown or to a disturbance resulting from an inaccurate planning like an over- or underestimation of the processing time. On the other hand, the content of an external disturbance can be either informative or performative. An example for an external disturbance with informative content is the disturbance associated with a rescheduling of the delivery of raw material. Performative external disturbances correspond to requests for the addition, deletion or modification of the planned schedule.

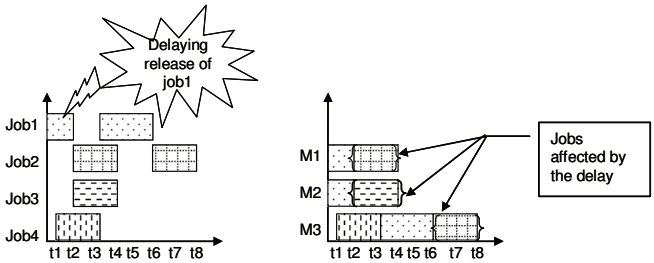


Figure 8.3: Scenario illustrating context of postponing a job release

The source and content of a disturbance are used to derive its context, which captures jobs and resources directly affected by the disturbance, as illustrated in Figure 8.1. To illustrate the context of a disturbance, consider the example depicted in Figure 8.3, in which four jobs are scheduled on three machines. A delay in releasing job1 affects machines M1, M2, and M3. By inspecting M1, M2, M3, the jobs affected by this event can be identified which are job2, and job3 in this case. Note that job4 is not affected by postponing job1 since it is scheduled ahead of it on M3. Therefore, job2 and job3 in addition to M1, M2, and M3 are captured by the context of this disturbance.

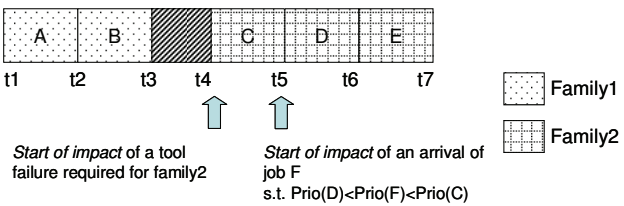


Figure 8.4: The start of impact for different scenarios

Modeling the context depends on specifying the start of impact. This represents the effective point in time at which the disturbance starts to cause an impact on the planned schedule. To illustrate, consider the example in Figure 8.4, where jobs of two different families are scheduled on a certain machine. Two possible scenarios are illustrated with the corresponding start of impact. The failure of a tool required for processing jobs of family2 at t1 causes an impact starting from t4. In the other scenario, the incoming of a job F with a priority falling between the priorities of C and D has a start of impact on this machine at t5. A disturbance may be associated with several points in time signaling the start of its effective impact on the several allocated resources.

Having specified the start of impact of a disturbance, the time period between the current time and the earliest start of impact can be calculated to indicate the time period in which an action

should be taken. This time period is referred to as the tolerance of the disturbance[5]. In other words, the tolerance is regarded as an estimation of the urgency of handling the disturbance and can thus aid in controlling the investigation of the action alternatives. The smaller the tolerance, the higher the urgency would be and the less action alternatives should be investigated. Some of the events have almost no tolerance because their detection is only possible after their occurrence and they have a direct start of impact once they occur. An example for this type is the over or underestimation of processing time. Although the detection of other disturbances such as a machine breakdown is also not predictable, these disturbances can have larger tolerance based on the current schedule of the concerned machines. For example, a failure at a machine which is waiting to process a job that will be released in a few hours is not the same as a failure of a machine that is currently processing jobs. Intuitively, the former disturbance has a larger tolerance than the latter.

8.2 Disturbance Handling Method

The main challenge in repairing the schedule in reaction to disturbances lies in the distributed representation of the generated schedule. The lack of a centralized representation of the schedule necessitates carrying out the schedule repair to a partial representation of the schedule under uncertainty about possible conflicts or reduction in efficiency that can result to other parts of the schedule. To allow for disturbance handling under this distributed representation, in a way that yields acceptable efficiency of the repaired schedule at the global level within allowable time limits, a disturbance handling method is proposed. As illustrated in Figure 8.5, the proposed method consists of five steps. These steps are explained in the rest of this section.

8.2.1 Identifying the Problem

Upon the occurrence of a disturbance, the associated context has to be derived to identify the jobs directly affected by the disturbance. Based on the start of impact associated with the context, the tolerance of the disturbance can be calculated. As an indicator for the urgency of handling the disturbance, the tolerance of a disturbance governs the action alternatives investigation, performed in the next step.

8.2.2 Investigating Action Alternatives

In handling a disturbance, different action alternatives are possible. These action alternatives are analyzed in Table 8.1 and Table 8.2, for internal and external disturbances respectively[6]. Even for the disturbance types for which a single action is listed, such as the case of inserting a job

[5] For simplicity, the time taken in repairing the schedule is not considered.

with a higher priority, several action alternatives exist, corresponding to the different available resources which can be allocated to the incoming job. It is worth noting that for some disturbances the best handling action is just to ignore the event and to continue with executing the planned schedule. In this case, the disturbance handling procedure has to be terminated, as captured in Figure 8.5. This is only possible in case of a disturbance with an informative content, for which the impact of doing nothing is less than that of taking a handling action. An example for this is an underestimation of the execution time of a certain job, which yields a small deviation from the planned schedule. In investigating potential actions, the impact of every action on the schedule, including no action, is to be calculated, as will be detailed in section 8.3.

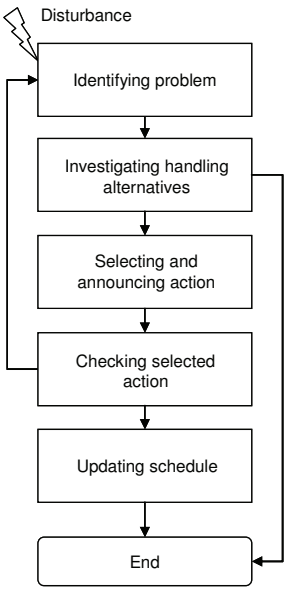


Figure 8.5: The disturbance handling method

8.2.3 Selecting and Announcing Action

The aim of this step is to find out the action that yields the best quality of the updated schedule. Due to the distributed schedule representation and the lack of a centralized problem solving entity, repair actions are selected based on limited local knowledge under uncertainty about the effect of the selected action on the overall performance. This problem is tackled by estimating

[6] In analyzing the action alternatives, the disturbance classification of the taxonomy of Figure 8.2 is used.

the expected impact of the local action on the global performance. In this way, selecting the action with the least relative impact results in the best relative performance at the global level. Following the action selection, the selected action is announced to the affected job agents. In case of adapting an integrated schedule, the transportation schedule is updated according to the repaired production schedule. The transportation resources are not informed directly by the production resources but rather indirectly through the job agents, in a similar procedure to the one used for generating a transportation schedule.

Table 8.1: Alternative actions for handling internal disturbances

Blocking disturbances	**Estimation inaccuracy**	
	Over-estimation	Under-estimation
Hold execution up to next resumption Forward to other resource(s)	Do nothing Advance affected jobs	Delay affected jobs Forward affected jobs to other resource(s)

Table 8.2: Alternative actions for handling external disturbances

Informative content	**Performative content**			
	Add request	Delete request	Modify request	
			Quantity	Priority
Hold execution up to availability of resource Forward to other resource(s)	Insert new job	Delete Delete and advance affected jobs	Modify quantity Forward request to other resource(s)	Move target job on same resource Allocate target job on other resources

8.2.4 Checking Selected Action

This step is necessary to ensure that the repair action taken by the agents representing resources and operations, out of their partial views of the schedule, does not result in conflicts to the schedules of other agents. By conflicts, we refer to violations of predefined constraints. These constraints correspond to the precedence constraints between the individual job operations and the collective real-time constraints for a given job group, such as the constraint that a given group has to finish execution by a given deadline. Every concerned agent performs a consistency check after being notified with a repair to its schedule, coming from another agent. Should the agent identify a conflict resulting from the proposed schedule repair, the identified conflicts is reported in terms of a disturbance, whose content formulates the desired delay of advance of the part of the schedule causing the conflict, as depicted in Figure 8.5 with the feedback loop from the action announcement step.

8.2.5 Updating Schedule

At this step, the selected action is simply applied to the schedule. Agents, proposing the repair action, commit the selected action in case no violation is reported from the notified agents, within a predefined timeout.

8.3 Extending the Agent-based Approach with the Disturbance Handling Method

This section addresses the details of extending the agent-based approach with the presented disturbance handling method. The adoption of the method in the agent-based approach is illustrated graphically in Figure 8.6, by capturing the added functionalities to the agents at the different levels of the architecture. In what follows, the added functionality at each of the four layers of the architecture is explained in details. Afterwards, the extension to the resources' proposal is explained.

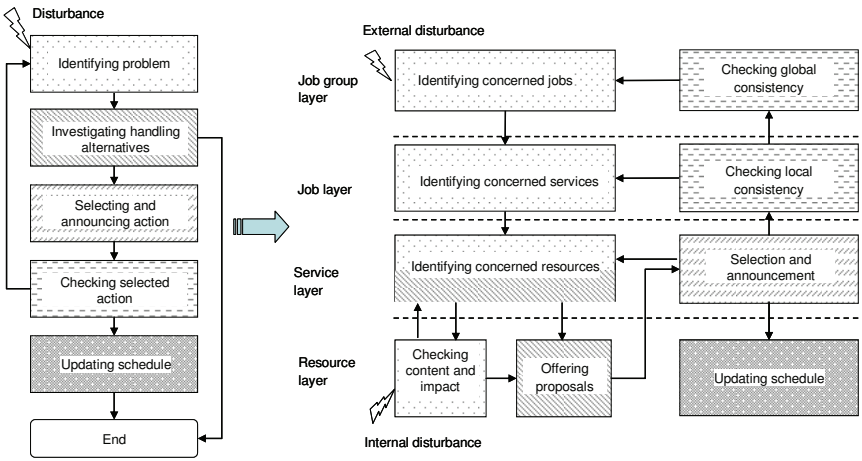


Figure 8.6: The adoption of the disturbance handling method in the agent-based approach

8.3.1 Extending Agents at Job Group Layer

Job group agents undertake the interaction between scheduling and planning. External disturbances (See Figure 8.2) are conveyed to the concerned job group agent. Consequently, a job group agent carries out partial identification of the problem, which aims at finding out the concerned job agents and passing on the disturbance's content to them.

In addition to participating in the problem identification, job group agents have to check their constraints, whenever they are notified with a schedule repair causing a change to their local schedules. Performing a consistency check can result in identifying a violation in a predefined constraint. For instance, repairing the schedule by delaying a job belonging to the group can result in violating the deadline for delivering the entire group. In this case, the job group agent reacts by formulating this violation as a new disturbance, whose content is forwarded to the concerned job agents to be propagated to the concerned resources. This is graphically depicted in Figure 8.6 by the feedback arrow from the checking functionality to the problem identification functionality.

8.3.2 Extending Agents at Job Layer

Similar to job group agents, the role of job agents in disturbance handling is mainly to inform the concerned agents at the lower levels of the architecture about the occurrence of a detected disturbance. In addition, they have to check that these proposals ensure the required consistency to the local schedule and to the group's schedule. To fulfill this role, job agents participate in the disturbance handling process by passing on disturbances delivered from their job group agents to the concerned operation agents. In addition, a consistency check for ensuring the fulfillment of the local constraints is necessary. This consistency check results in iterating back to the problem identification, in case a constraint violation is identified, or proceeding to the global check at the job group level, otherwise (See Figure 8.6). The feedback to the problem identification functionality is also necessary in case of the integrated scheduling mode, to notify the concerned transportation resources with updates in the production schedule dictating an adaptation to the transportation schedule.

8.3.3 Extending Agents at Service Layer

Operation agents are involved in the problem identification of external disturbances. As previously explained, external disturbances are detected by the concerned job group agents and passed on until they are received by the concerned operation agents. Operation agents react to these disturbances by identifying the resource agents concerned with the handling action and contacting them to prepare schedule repair proposals. On the other hand, internal disturbances are captured from the resource agents representing the source of the occurring disturbance. The role of service agents in this case is to find a way compensating the problem of the resource initiating the handling request through other resources that currently support the concerned service. Accordingly, the identification of resources in case of an internal disturbance corresponds to the step of investigating action alternatives, which justifies the graphical representation of this functionality with both the filling pattern of the problem identification and that of the investigation of action alternatives in Figure 8.6.

In addition, both operation and transportation agents cooperate in disturbance handling by evaluating the proposals offered from their resource agents and select the one with the least impact on the affected jobs and resources. Having selected an action, the new schedule resulting from applying this action is announced to the agents representing the affected jobs. This corresponds to the selection and announcement of the repair action in the presented disturbance handling method, as depicted in Figure 8.6.

8.3.4 Extending Agents at Resource Layer

Due to their 1-1 mapping with the physical resources, resource agents are equipped with monitoring capabilities through interface channels with their corresponding resources (See section 5.5). Through these channels, resource agents are continuously informed about the current status of their resources and maintain this status in their self model. Upon receiving a notification of a disturbance, like a machine breakdown, the resource agent first figures out whether the disturbance can be ignored or not. This decision is based on the type and impact of the disturbance. As previously mentioned, disturbances for which no action must be taken are those with informative content and an impact within allowable range, specified by a given threshold. Therefore, resource agents are extended with a functionality for checking the content and the derived impact of the disturbance, which corresponds to the problem identification step of the disturbance handling method (See Figure 8.6).

In case an action is necessary, another decision is to be taken concerning whether this action involves the participation of peer resource agents, currently providing the same service. This decision is taken based on the tolerance of the disturbance. If the tolerance of the disturbance falls below a given threshold, a handling action has to be taken with high urgency and thus a handling proposal is prepared locally and forwarded to the operation or transportation agent to be announced to the concerned agents at the upper layers. If, on the other hand, the tolerance allows for a more thorough investigation of the handling action, the identification functionality inside the concerned agent at the service layer is invoked. This informed agent calls the corresponding resource agents to prepare handling proposals, as in Figure 8.6. In addition, resource agents, once they receive an acceptance from the agents at the service layer, update their local schedules by applying the proposed action. Similarly, other agents along the upper three levels of the architecture confirm their local schedules.

8.3.5 Extending Resource Proposals with Disturbance Handling Costs

In addressing schedule generation, it has been shown that resource proposals for both the production and the transportation schedules entail a scheduling cost, which represents a decisive factor in the selection step. Similarly, in updating the schedule in reaction to disturbances, costs

have to be estimated and included in the proposals of the resources. However, the costs in case of disturbance handling have to account for the estimated impact not only on the resource itself but also on the affected jobs, which are to be rescheduled by the handling action. In what follows, the calculation of the rescheduling costs of the production and the transportation schedules is highlighted.

8.3.5.1 Costs of Updating Production Schedules

The costs of updating a production schedule correspond to the impact of a handling action on the local schedule of the concerned resource. The impact $\lambda(A,C)$ of a certain action A applied to a context C is estimated by accounting for both the utility of the selected action, measured in terms of the added setup, and the stability in terms of the deviation of the repaired schedule to the original schedule. This is based on a weighted sum consisting of an estimate of the reduced machine utilization of applying action A to context C, $RMU(A,C)$, as a measure of the action's utility, and an estimate of the associated stability $RS(A,C)$. As in (8.1) these two factors are added together using a multiplication factor, w_u, that represents the weight of the utility relative to the stability.

$$\lambda(A,C) = w_u \cdot RMU(A,C) + (1 - w_u) \cdot RS(A,C) \tag{8.1}$$

$RS(A,C)$ is calculated by either the number of affected jobs or the weighted deviation suggested in [RFK04]. On the other hand, $RMU(A,C)$ is calculated based on the difference between the unutilized machine time in the context before and after applying action A.

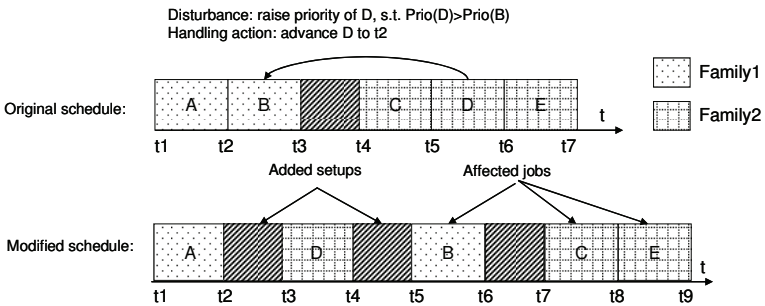


Figure 8.7: Illustration of the calculation of impact

Figure 8.7 depicts the context associated with a disturbance resulting from increasing the priority of job D. In this case, one of the action alternatives is to advance the start time of D on the same machine. Since the new allocation of D lies between two other jobs from the same family, it results in two added setups. For simplicity, it is assumed that the setup time as well as

the processing time is normalized to one time unit. In this case, the reduced stability amounts to 3, corresponding to the number of affected jobs and the reduced machine utilization amounts to 2, corresponding to the 2 added setups.

8.3.5.2 Costs of Updating Transportation Schedules

Besides the transportation time taken to deliver the requested job from the source to the destination, the transportation costs in case of schedule update include an estimation of the impact on the jobs affected by the selected action. The total cost is calculated as follows.

$$C = w_D \cdot D + (1 - w_D) \cdot T \tag{8.2}$$

While w_D is a predefined weight, D is an estimation of the delay caused to the already scheduled jobs and T is the additional transportation time associated with fulfilling this request. While T represents a measure of the utility, D is a measure of the stability. A good compromise between the utility and the stability of the repaired schedule can be achieved by configuring w_D.

8.4 Investigation of the Applicability of the Disturbance Handling Method

In this section, the applicability of the proposed disturbance handling method is investigated by considering disturbances, both internal and external, and explaining how they are handled by the proposed method.

8.4.1 Handling Internal Disturbances

In illustrating the reaction to internal disturbances, breakdowns of both the processing resources and the material handling resources are considered.

8.4.1.1 Handling Breakdowns of Processing Resources

With processing resources, we refer to failures hindering a machine to carry out a certain operation, for which it is configured. This includes failures to the machine itself or to the tools with which the machine is currently equipped. Both failures are detected by the machine agent representing the machine. The occurrence of these types of disturbances causes a message to be sent to the corresponding resource agent through the interface with the shop floor control. Once this message is received by the agent, the self model is updated to reflect the current status of the physical resource. The current status is kept in the self model as a base for the agent's decisions and its readiness to participate in the schedule generation.

To demonstrate the application of the disturbance handling method in reaction to a processing resource failure, consider the scenario depicted in Figure 8.8. Four jobs associated with two different part families are scheduled on three CNC machines performing the same operation. Before starting the execution of these jobs and while exchanging tools to configure CNC3 with the required part families, a tool required for part family1 breaks. This event is captured by M3, the agent representing CNC3, and triggers the execution of the disturbance handling method, as illustrated in Figure 8.8. The sequence diagram in the figure shows the interactions among the agents upon the detection of the disturbance by M3. The activation boxes of the sequence diagram are annotated with the number of the corresponding step of the disturbance handling method. The steps are listed under the sequence diagram to facilitate their traceability.

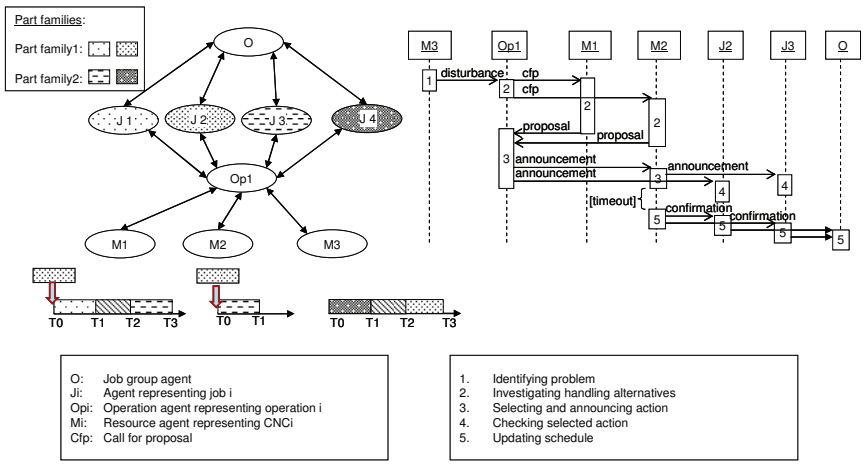


Figure 8.8: A scenario illustrating the handling of a broken tool

M3 reacts to the event by checking the context and deriving the impact of the disturbance. By considering its local schedule kept in its self model, M3 determines the start of impact, which is in this case the start of the first job belonging to part family1, i.e. job2. As depicted in Figure 8.8, the context encompasses only J2 in this case since no other job is scheduled after it at CNC3. In light of the identified context, the impact is derived, which is a delay to job2. Due to the lack of information about the point in time at which the broken tool can be exchanged with a working one, the disturbance cannot be handled locally and it has to be forwarded to the operation agent Op1 (See Figure 8.8).

Having received a notification of the disturbance, Op1 initiates the second step in the disturbance handling method, namely the investigation of handling alternatives, by figuring out the resource agents that are currently configured with the tools required for part family1, to request them to compensate for the failure. Accordingly, it sends calls for proposals to all the

identified agents, as captured by the sequence diagram in Figure 8.8. A call for proposal to handle a disturbance causes the receiving resource agents to engage in investigating the action alternatives. Each one of these agents prepares a proposal consisting of the offered start time of the rescheduled job and the associated costs. Assuming that job2 has got a higher priority than job1 and job3, if job2 is to be rescheduled at CNC1, it would be allocated ahead of the already scheduled jobs, i.e. job1 and job3. Consequently, the new start time would be T0 and the associated costs would correspond to the delay of 2 jobs. No extra setup is necessary since the part types of job2 and job1 belong to the same part family[7]. Assuming a predefined weight, w_u, of 0.25, the stability of the modified schedule is to be given a significance, three times higher than that of the utility, i.e. to be weighed with 0.75. Consequently, the costs amount to 1.5 resulting from multiplying 0.75 by 2, assuming that the reduced stability is to be estimated based on the number of delayed jobs. On the other hand, the proposal offered by M2 entails a starting time of T0 and costs amounting to 1, resulting from the addition of two factors. The first factor represents the multiplication of 0.25 by 1 (one extra setup required to switch from part family1 to part family2). The second factor corresponds to the multiplication of 0.75 by 1 (one delayed job).

These proposals are gathered by Op1 and evaluated based on the offered start time and the associated costs. Since the offered start times are the same, the costs represent the decisive factor in the selection and hence M2 is selected. Upon the selection of a handling action, the action has to be announced; therefore, Op1 announces its selection by notifying the selected resource agent, M2, and the agent representing the job to be rescheduled, J2. M2 informs in its turn the affected job agents, J3 in this case. As illustrated in Figure 8.8, this announcement causes J2 and J3 agents to carry out the fifth step of the disturbance handling method by checking whether their internal constraints would be violated by the announced change. Similarly, the job group agent is notified with the change. In the considered example, no constraint is violated and accordingly, no feedback is given from the contacted job agents. After the elapse of a predefined time-out, M2 assumes that the involved parties are satisfied with the announced action and commences the sixth and final step, in which the schedule is updated. The schedule update takes place by applying the announced action to the local schedule of M2 and sending a confirmation to J2 and J3, which likewise update their local schedules.

8.4.1.2 Handling Breakdowns of Material Handling Resources

In this section, the adaptation of the schedule in reaction to an AGV breakdown is illustrated based on the scenario captured in Figure 8.9. The considered FMS consists of two FMCs connected with two work-in-progress buffers (See Figure 8.9 (a)). While FMC1 contains three AGVs, FMC2 contains only two. An AGV can only move around within the boundaries of its

[7] For simplicity, the minor setup time associated with the switch from a part type to another from the same part type family is ignored.

encompassing FMC. For transporting workpieces across FMCs, AGVs deliver their load to the input buffer of the destination FMC, where the delivered workpieces can be moved, by another AGV belonging to this FMC, from the input buffer and delivered to the destination machine[8].

The agent-based representation of the material handling resources, together with a job agent for which a transportation schedule is to be generated, is depicted in Figure 8.9 (b). In the same figure, the production schedule of the requesting job (J1) is captured. According to the production schedule of J1, workpieces have to be transported from M2 to M5. Therefore, the generation of the transportation schedule results out of the interaction between J1 and Tr1, i.e. the transportation agent responsible for the FMC containing the source machine, M2. Since the destination machine, M5 is located in FMC2, the interaction between Tr1 and Tr2 is necessary to arrange for bringing the workpieces of J1 from M2 in FMC1 all the way to M5 in FMC2.

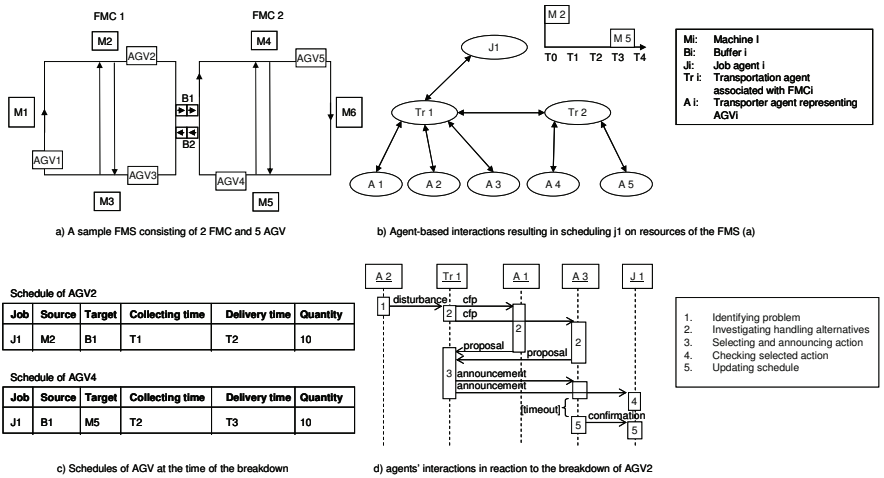


Figure 8.9: A scenario illustrating the handling of an AGV breakdown

The transportation schedule that resulted from these interactions is tabulated in Figure 8.9 (c). Two AGVs are allocated for J1, AGV1 for transporting the workpieces from the source machine to the input buffer B1 of FMC2, and AGV4 for moving the workpieces from B1 to M5. Every AGV is allocated 10 palettes of workpieces, representing the total quantity of J1.

Before the release of job J1, AGV2 breaks down. The sequence diagram capturing the interactions among the concerned agents in reaction to this disturbance is sketched in Figure 8.9 (d). The occurrence of the breakdown is reported to the agent representing the broken AGV, A2. Assuming that the breakdown period is not known, the disturbance cannot be handled locally and thus a notification is forwarded to Tr1. Tr1 reacts by retrieving the resource agents

[8] For the sake of illustration, the load and unload stations are ignored.

representing the AGVs that can compensate the breakdown of AGV2 and requesting them to prepare handling proposals. This represents the start of step 2 in the disturbance handling method, namely the investigation of handling actions. This step is carried on by the contacted resource agents, A1 and A3 in the illustrated example. Each of these resource agents responds with a proposal comprising new collecting time, delivery time and costs, whose preparation was explained in 8.3.5.2.

The arrival of proposals at agent Tr1 triggers the execution of step 3, i.e. the selection and announcement of a handling action. Assuming that A3 provides the best relative proposal, an announcement is sent from Tr1 to A3 informing it with the selection of its proposal. Similarly, J1 is informed with its modified schedule. Assuming that no constraint is violated, the schedule is updated by A3 after the elapse of a predefined timeout and this is confirmed with J1, for the latter to update its schedule too.

8.4.2 Handling External Disturbances

To demonstrate the reaction to external disturbances, the incoming of a higher priority job based on the scenario depicted in Figure 8.10 is considered. A job group consisting of two jobs is scheduled on three machines performing two operations. The agents involved in this scenario along with their interactions that led to scheduling J1 and J2 are depicted in Figure 8.10 (a).

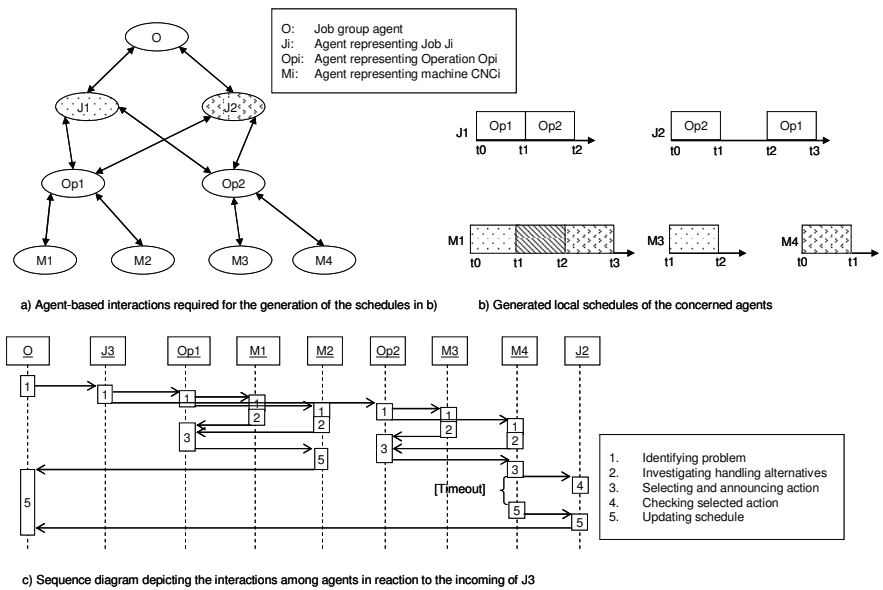


Figure 8.10: A scenario illustrating the handling of a higher priority job

The distributed schedule, from the machine and the job perspectives, is sketched in Figure 8.10 (b). Since machine CNC2 is not allocated to any job, no schedule is displayed for it. A setup time is planned between jobs J1 and J2 at machine CNC1, since they belong to two different part families.

Before releasing the scheduled jobs, a rush order for scheduling job J3 with a higher priority than J1 and J2 arrives. This new job belongs to the existing job group and hence its incorporation in the schedule is initiated by informing the agent O, representing this group. This agent reacts to the incoming disturbance by interacting with the concerned agents according to the sequence diagram, sketched in Figure 8.10 (c). The first step is to identify the problem caused to the schedule by this disturbance. This step starts with the agent O which checks the required action and instantiates an agent for the new job and informs it with the event. In its turn, the new job agent J3, finds out the required operations by looking up the plan of its associated part type from the environmental model.

According to the retrieved plan, agents representing the required operations, Op1 and Op2 in the considered example, are contacted. These agents have to carry on the problem identification step by propagating the event to the concerned resource agents. At the resource level, the exact problem caused to the schedule can be identified by specifying the disturbance context. Based on the specified context, the impact of the handling action can be derived and included in a proposal, which corresponds to the start of step 2, i.e. the investigation of action alternatives. The two operation agents receive the resource proposals to select the best out of them. While Op1 selects the proposal of M2, Op2 selects that of M4, assuming that J3 causes no extra setup for M4 because it belongs to the same part type family of J2. The selected proposal is announced to the concerned resource agents, which in their turn announce the resulting modification to the affected jobs.

Announcing a schedule modification to a job agent triggers the checking of internal constraints, as the case with job J2, whose Op2 would be delayed by one time unit. Since this delay causes no violation to the internal constraints of J2, the schedule is updated by M4 and the new schedule is notified to J3 which reports the modified schedule to the job group agent O.

In this chapter, the schedule repair mode has been addressed. The different manufacturing disturbances causing the schedule to be repaired have been analyzed and classified. Out of this analysis, a method for repairing distributed schedules has been proposed. Required extensions to the different agents to incorporate this method have been explained and illustrated. The applicability of this schedule repair method in handling the different types of disturbances has been illustrated.

9 Realization of the Concept

The proposed agent-based FMS scheduling concept has been realized in form of a scheduling framework. In this chapter, details are given about the developed framework. Section 9.1 gives an overview on the framework and its internal structure. Section 9.2 describes the graphical user interface and illustrates main use cases of the framework. In section 9.3, the emphasis is laid on details concerning the developed agents. Section 9.4 explains the integration of the framework into its environment. As part of the environment, a simulated control has been developed. Details of this simulated control is given in section 9.5.

9.1 Overview on the Scheduling Framework

The proposed concept has been realized by developing GADWAL: an inte**g**rated **A**gent-base**d** frame**w**ork for dyn**a**mic schedu**l**ing. It is a generic agent-based framework that can be easily configured for any FMS. The internal structure of the framework and its interfacing channels to the external components are captured in Figure 9.1. The internal structure of GADWAL consists of three software components: the agent-based subsystem, representing the core of the framework, a graphical user interface (GUI) and an interface to the environmental model. GADWAL provides communication channels to two bi-directional external components: the simulated control and the environmental model.

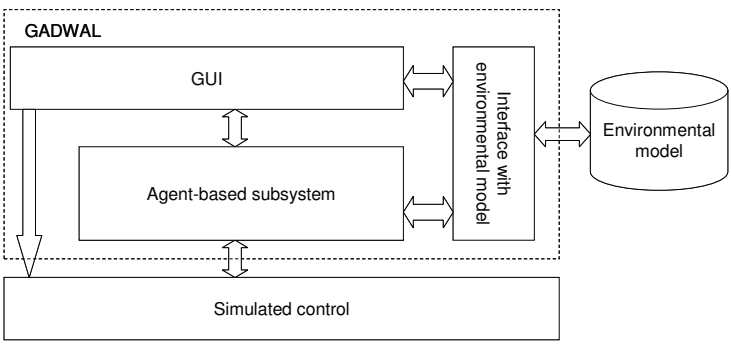


Figure 9.1: The agent-based scheduling framework integrated with its environment

9.2 Use Cases of GADWAL

The graphical user interface consists of configuration, scheduling, visualization and performance evaluation functionalities.

9.2.1 Configuration

This functionality allows the user to configure the framework for a specific FMS based on a few easy steps. In addition, it allows for configuring the scheduling process through setting the scheduling mode, the optimization objectives and optimization-related parameters. Figure 9.2 shows the steps necessary for configuring the framework to a different FMS. In the first step, the resources of the FMS along with their interdependencies are defined. Resources that are relevant to scheduling are specified and their attributes are fed into the framework.

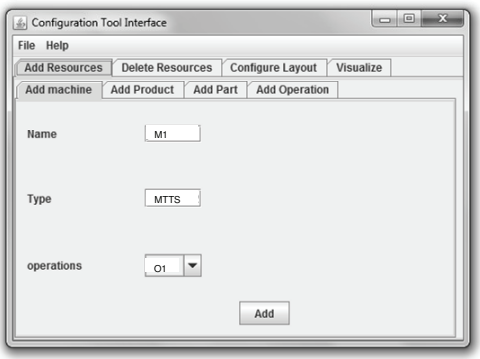


Figure 9.2: Configuring the framework to a new FMS

In addition, the spatial layout of the shop floor and the relative locations of the previously defined resources, are supplied in the form of an XML file. The structure of this file is identified in a document type definition (DTD), as captured in Figure 9.3. For describing the physical layout of a shop floor, the identification of the physical locations of the machines and the work-in-progress buffers is necessary. In addition, the initial position of the AGV is also required, which is used to derive the corresponding FMC and hence transportation boundaries of the AGV. The machines are defined in terms of their input and output buffers which are connected to the transportation paths through junctions. Besides configuring the framework to a specific FMS, the GUI allows for the configuration of the scheduling process through setting the scheduling mode, the optimization objectives and the optimization-related parameters, such as the predefined weights.

9.2.2 Scheduling

Through the graphical user interface, planned jobs can be supplied to scheduling and the schedule generation process can be initiated. Jobs can be supplied to the agent-based sub-system either interactively, by defining the necessary attributes in a successive way, or based on an

input file, containing the set of jobs with their associated attributes, as shown in Figure 9.4. While the former way suits the sequential scheduling mode, the latter is appropriate for the simultaneous scheduling mode. The schedule generation is initiated by clicking a button. In addition, disturbances causing the adaptation of the generated schedule can be defined and triggered from this functionality.

```
<!ELEMENT Shopfloor (Elements, AGVs)>
<!ELEMENT AGVs (Agv)*>
<!ELEMENT Agv EMPTY>
<!ELEMENT Junction EMPTY>
<!ELEMENT InBuffer EMPTY>
<!ELEMENT OutBuffer EMPTY>
<!ELEMENT Elements ((InBuffer|OutBuffer|Junction)*)>
<!ATTLIST Agv
Id CDATA #IMPLIED
Junction IDREF #REQUIRED
Speed CDATA #REQUIRED
Capacity CDATA #REQUIRED>
<!ATTLIST Junction
```

Figure 9.3: A snippet of the document type definition of the shop floor description

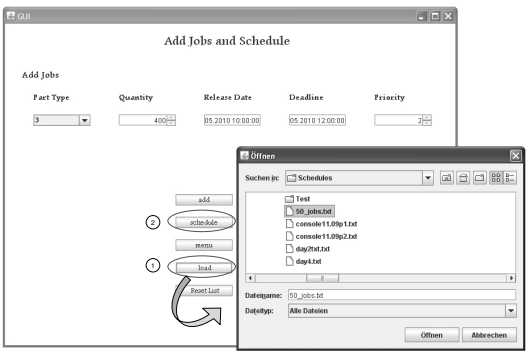


Figure 9.4: Scheduling a set of jobs saved in a file

9.2.3 Visualization

The visualization functionality enables the required transparency by giving an insight to the user into not only the generated schedules but also the partial results reached by the agents. The generated schedule is represented both textually and graphically, in form of a Gantt chart, and from the resource and the job perspectives. The partial results correspond mainly to the schedules resulted from the different intermediate generations, in case of the GA-based optimization. These partial results are significant in evaluating the performance, as will be discussed in the next section. For this purpose, file management functionalities for the generated

schedules are supported. These enable the loading and saving of the schedules generated so far as well as regaining the current schedule through the compilation of the decentralized schedules attained from the concerned agents into a comprehensive and coherent schedule.

9.2.4 Performance Evaluation

This functionality is mainly intended for research purposes and for guiding the decision makers in the factory. It helps the user in assessing the performance of the generated schedule according to the different evaluation criteria, such as the makespan or the resource utilization. Running a performance test over a set of GA-generations is supported through generating a file containing the evaluation results of the partial schedules along the entire set of generations. By plotting these results in form of a graph, a quick overview can be attained about the agent-based optimization over the different generations. In addition, this functionality supports the assessment of the agent-based reaction to disturbances by comparing the schedules before and after the occurrence of the disturbing events.

9.3 Realization of the Agent-based Subsystem

The proposed agent-based scheduling concept has been realized in Java under the Whitestein LS/TS platform [Whit06]. Each agent type is realized as a separate package consisting of the source code corresponding to the agent main functionalities and the associated tasks. Agents communicate together using the features supported by LS/TS, which enables agents to register themselves by the provided services and to be invoked by other agents by calling a directory facilitator, similar to the concept of the yellow pages. The instantiation of agents occurs dynamically, as described in section 9.2.1.

9.4 Integration of the Framework into the Environment

The environment of GADWAL corresponds to the planning, the shop floor control and the environmental model.

9.4.1 Interface with Planning

The interaction with planning is enabled through message passing. In addition, every agent is equipped with message handlers for dealing with possible messages from planning. Currently, the outcome of planning is fed into scheduling through the user interface, as demonstrated in section 9.2.

9.4.2 Interface with Shop Floor Control

GADWAL provides interfacing channels with simulated shop floor controls based on sockets. These channels allow for the bidirectional dynamic data exchange between agents and the simulated control.

9.4.3 Interface with Environmental Model

The environmental model is realized as a relational database. Bidirectional access to the environmental model is abstracted from the low level details of the access to relational databases by using hibernate, an object-relational mapper. The structure of the environmental model is predefined once by constructing the required tables. For each of these tables, a data access object (DAO) has been written and a hibernate mapping has been created (See Figure 9.5), to establish the link between the relational database table and the DAO. In addition, a number of access services have been coded in java providing agents with easy access to the environmental model.

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
   "-//Hibernate/Hibernate Mapping DTD//EN"
   "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>

    <class name="fmsControl.dao.Part" table="PART">

        <id name="PartNo" column="PART_NO" type="java.lang.Integer"/>
        <property name="minorSetup" column="MINOR_SETUP" type="java.lang.Integer"/>
        <property name="familyNo" column="family_no" type="java.lang.Integer"/>


    </class>

</hibernate-mapping>
```

Figure 9.5: The hibernate mapping of the part type entity

9.5 Simulated Control

Plant Simulation from Technomatix is used for modeling FMSs and simulating their shop floor control. To facilitate the testability of the framework, a configuration tool for FMS simulation has been developed. The task of this configuration tool is to create simulation models with their embedded control for given FMS descriptions. FMS descriptions are encoded into the XML format and are generated during the configuration step, illustrated in 9.2.1. Figure 9.6 presents the internal architecture of this configuration tool. It consists of a components library and an FMS modeling engine. The components library consists of modeling objects with their embedded control. These components correspond to the following resource types: loading stations, unloading stations, CNC work stations, AGV, work-in-progress buffers and connection

lines. Every component is modeled by an object class that inherits from a built-in class and is extended with the required control for FMSs.

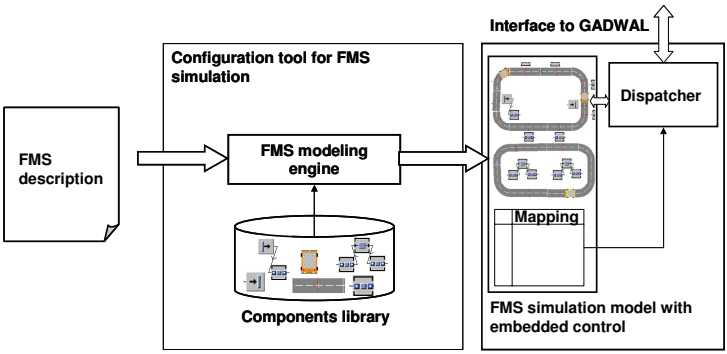


Figure 9.6: Simulation of a predefined FMS

The configuration tool generates a model, corresponding to the FMS description, and a mapping list for establishing the link between the identifiers of the modeled resources and those of the corresponding agents. The model is accompanied with a bidirectional dispatcher which realizes the dynamic data exchange through sockets between the modeled resources and their corresponding agents.

This chapter has described the development of a scheduling framework which realizes the proposed concept. The structure of the developed framework has been described. The different use cases supported by the framework have been explained and illustrated. Finally, implementation details for realizing the different components of the framework and integrating it with the environment have been given.

10 Evaluation based on an Application Scenario

The proposed concept has been evaluated by modeling an IBM test line, based on data reported in [Witt90]. Before getting into the details of the evaluation, the considered FMS is described and its modeling in the agent-based framework is briefly presented, by focusing on the simulated control model and the instantiated agents. Following this introductory presentation of the FMS model, evaluation results are discussed by considering the different scheduling modes, illustrated in section 4.3. The evaluation is mainly based on conducting experiments aiming at the assessment of the requirements derived in chapter 2. The chapter concludes by investigating the degree of fulfillment to these requirements based on the presented evaluation results.

10.1 Overview on the Considered IBM Test Line

The test data have been collected from an IBM test line. This test line represents a huge FMS that consists of thirty one testers grouped into four families. Each tester family is capable of performing a specific test on a predefined set of card types. A total of ten different card types grouped into four families are to be tested on one or more testers according to their predetermined plan, described in Table 10.1. Tester types correspond to the required operations and are not governed by precedence constraints.

Table 10.1: Routings of the IBM test problem [Witt90]

	Tester type (quantity)			
Part type	MPTS(12)	GSAT(5)	MTTS(12)	CLIV(2)
1			▪	
2			▪	▪
3	▪			
4		▪	▪	
5			▪	
6			▪	
7	▪	▪		
8	▪			
9	▪			
10			▪	

The problem is characterized by major and minor setup times. Setup times depend just on the part type being switched to, as illustrated in Table 10.2. Each part family is associated with a major setup time which amounts to 10, 20, 30 and 40 minutes for part families 1, 2, 3 and 4 respectively. The goal is to maximize the throughput, which is defined as the total number of parts to be manufactured divided by the makespan. By assuming a fixed total number of parts

over a certain manufacturing period, the problem reduces to minimizing the makespan [Witt90]. In order to minimize the makespan, the setup time has to be minimized, which is only attainable by optimizing the sequencing and allocation sub-problems.

Table 10.2: Description of the supported part types – adapted from [Witt90]

Part type	Part family	Minor setup (min.)
1	1	5
2	2	10
3	1	5
4	3	5
5	3	10
6	4	5
7	1	10
8	4	5
9	4	10
10	4	10

10.2 Modeling the IBM Test Line

According to the data reported in [Witt90], a simulation model has been constructed in Plant Simulation, as described in section 9.5. Since the approach provided in [Witt90] belongs to the offline machine-centric approaches, details about material handling are disregarded. This has been dealt with by modeling the layout as depicted in Figure 10.1. The model consists of five FMCs: one main FMC and four other FMCs containing workstations. Workstations correspond to tester types and are distributed randomly in the four FMCs. Each workstation comprises an input and an output buffer for the temporary storage. The main FMC contains a source for generating workspaces and a drain corresponding to the unload station. Transporting workpieces from the loading station to the different workstations all the way up to the unloading station follows the schedule delivered from the agents and is undertaken by AGVs. In each cell, two AGVs are made available.

Agents modeling the IBM test line have been dynamically instantiated – as described in section 9.2 – (See Figure 10.2). The number of resource agents amounts to 31 machine agents, ten AGV agents, one material allocator agent corresponding to the load station and eight material stores representing the input and output buffers connecting the different FMCs with the main FMC in the center. At the service level, four operation agents are instantiated to represent the four test operations, listed in Table 10.1. In addition, five transportation agents are instantiated to represent the transportation services within the five FMCs.

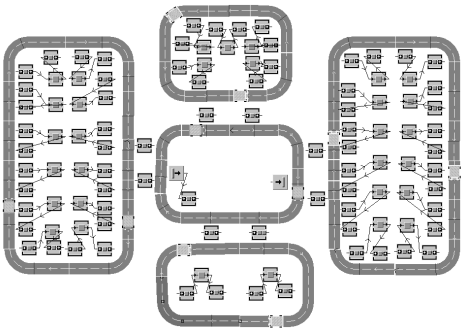


Figure 10.1: The simulation model of the IBM test line

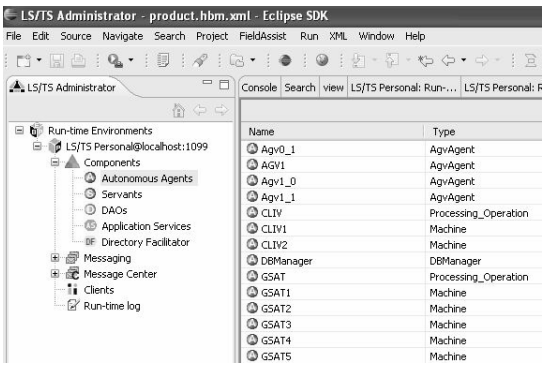


Figure 10.2: Subset of the dynamically instantiated agents displayed by LS/TS

10.3 Evaluation of Schedule Generation

In this section, the schedule generation mode is evaluated by first considering production scheduling followed by integrated scheduling.

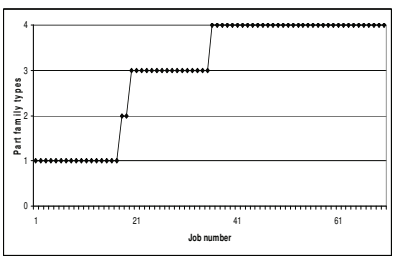
10.3.1 Evaluation of Production Scheduling

In evaluating the production scheduling mode, the framework was configured to adopt a machine-centric scope for scheduling. Experiments have been conducted to evaluate both the sequential and the simultaneous modes of production scheduling. The selection of one of these two modes was made automatically by the framework based on the number of requests received by operation agents within a predefined period of time.

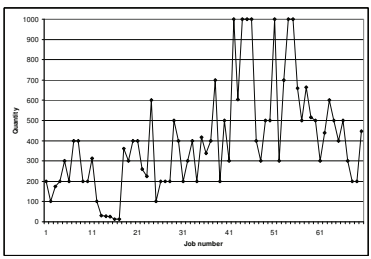
10.3.1.1 Evaluation of Sequential Scheduling

As previously illustrated, the sequential scheduling mode corresponds to generating production schedules under the sequential arrival of job requests. In the sequential arrival model, jobs have to be scheduled under uncertainty about the types and quantities of the job requests to come later. The experiments conducted in this section aim at estimating the influence of the stochastic nature of the incoming requests on the efficiency of the generated schedule. Since in [Witt90] a simultaneous job arrival model is assumed, data are presented in terms of the collective values of part types and quantities of the jobs to be manufactured over a whole day. Therefore, these data have been customized to suit the sequential job arrival model, by randomly dividing the total quantities on a given number of jobs.

To evaluate the effect of the stochastic nature of the scheduled jobs on the efficiency of the generated schedule, these jobs were arranged in two datasets, distinguished from each other in the job arrival sequence. In dataset I, jobs were grouped according to their part family types. The arrival sequence of jobs was determined based on their part families. This led to the sequence depicted in Figure 10.3(a), where variations in part family types between subsequent jobs are minimized. The associated quantities of these jobs are depicted in Figure 10.3(b). Dataset I represents the best case for scheduling where the changeover of the machines is minimized. For the generation of dataset II, the arrangement of the same set of jobs was shuffled in a way that features great alternation in part family types and quantities between subsequent jobs (See Figure 10.4).



a) Variations in part family types

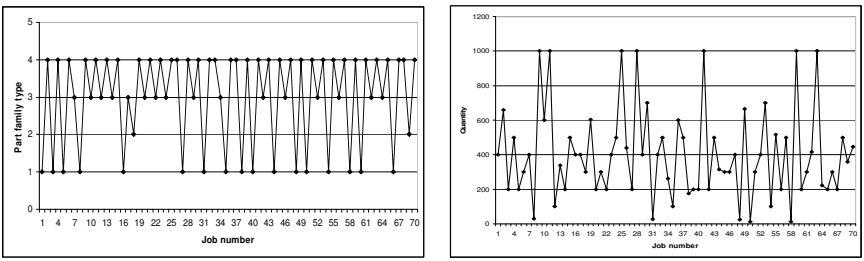


b) Variations in quantities

Figure 10.3: Variations in part types and quantities of dataset I

The two datasets were used to compare the performance of the proposed agent-based scheduling approach to the traditional priority-rules based approach with respect to the resistance to the stochastic nature of the planned jobs. The FCFS priority rule was applied for sequencing at the machine level since it represents the intuitive choice in case arriving jobs have to be instantaneously incorporated into the schedule. In applying FCFS, the entire set of machines for every requested tester type was allocated to the requesting jobs. The results are plotted in Figure

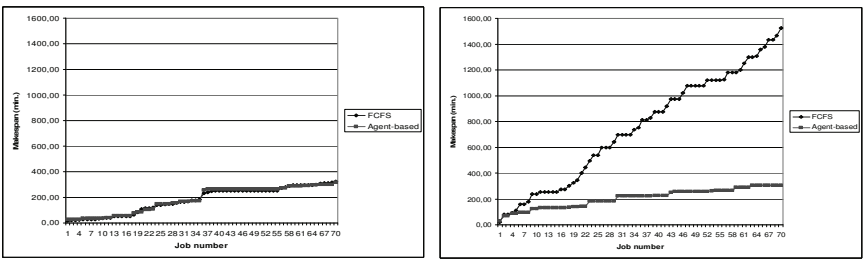
10.5 by comparing the performance of the conventional FCFS scheduling with the proposed agent-based scheduling, using the two heuristics described in chapter 7.



a) Variations in part family types

b) Variations in quantities

Figure 10.4: Variations in part types and quantities of dataset II



a) Test results of dataset I

b) Test results of dataset II

Figure 10.5: The effect of fluctuations in part types and quantities of the scheduled jobs on both the conventional and the proposed scheduling

In case the alternation in part types of subsequent jobs is minimized, the proposed agent-based approach results in a similar makespan to that of the conventional approach. This is reflected by the comparison of the makespan for both approaches using dataset I, as illustrated in Figure 10.5 (a). By introducing fluctuations between subsequent jobs, FCFS scheduling was found to yield significant increase in makespan due to the frequent changeover of the setup of the machines which increases the setup time and delays jobs. On the other hand, the proposed agent-based approach shows resistance to change. This is manifested in the performance of both approaches to the jobs of dataset II, as depicted in Figure 10.5 (b).

Compared to the lower bound derived mathematically in [Witt90], the performance of the proposed approach lies within 9.9% of the optimal makespan. Considering the centralized static scheduling based on a static model of job arrival, where all data about incoming jobs were determined a priori, a schedule within 3% of the optimal solution could be generated [Witt90].

In conclusion, under uncertainty about the types and quantities of the incoming job requests, the proposed agent-based scheduling was found to significantly outperform the FCFS scheduling and approach the near optimal solution resulting from the static scheduling, where job requests are deterministic.

10.3.1.2 Evaluation of Simultaneous Scheduling

The purpose of this evaluation was mainly to estimate the efficiency of the generated schedules and to investigate how far it affects the reactivity, measured in terms of the computational complexity. Different test experiments with various parameters have been conducted. The presented experiments were conducted based on the parameter configuration shown in Table 10.3.

Table 10.3: Parameter configuration for the conducted experiments

Parameter	Value
Parallelization factor (r)	2.5
Predefined weight (W)	0.25
Population size (At resource level)	10
Population size (At operation level)	50
Crossover rate	0.8
Mutation rate	0.02
Crossover type (At resource level)	One-point crossover
Crossover type (At operation level)	Two-point crossover
Diversification factor	0.25

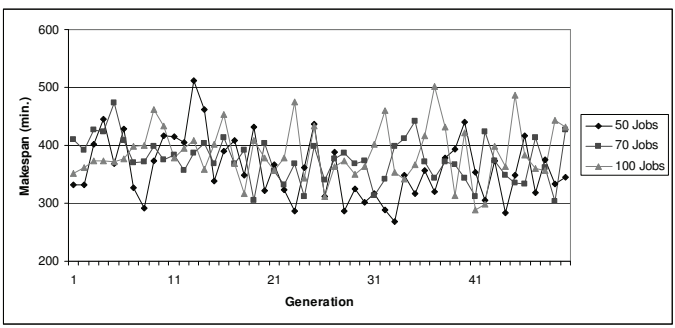


Figure 10.6: The effect of changing the number of jobs on the resulting makespan

The makespan was taken as a measure of efficiency. First, the aim was to investigate the effect of increasing the problem space on both the efficiency and the reactivity of the proposed algorithm. Part types and quantities corresponding to production orders from a typical manufacturing day were used to define datasets of different numbers of jobs, namely 50, 70 and 100 jobs. In other words, these datasets share the same optimal makespan but differ in the size

of the search space. The makespan of the generated schedules for the three datasets was investigated along fifty hierarchical GA generations. A hierarchical generation refers to the execution of the local evolution at the resource level followed by the execution of the local evolution at the operation level. Generations that resulted in invalid schedules were simply ignored and not counted. While the first dataset with 50 jobs resulted in an optimal schedule after 33 generations, the other two sets resulted in schedules with a deviation from the optimal makespan ranging from 7% to 13% in the span of the 50 generations (See Figure 10.6). Evidently, increasing the solution space most probably leads to delaying the convergence. However, the computational complexity is low enough to allow for a larger number of generations. On average, one hierarchical generation was found to take around 0.3 seconds on a 2 Duo 2.0 GHz processor.

In the previous experiment, the optimization was pursued by evolving one generation at the resource level followed by another generation at the operation level. In this experiment, the objective was to investigate the influence of increasing the number of generations at the local resource level on the makespan of the resulting schedule. Figure 10.7 illustrates the results of applying three different runs for dataset II, with 5, 7 and 11 generations at the resource level. Initially, increasing the number of local generations was found to expedite the convergence. With a local optimization of 5 generations, a near-optimal schedule could be generated at the 27^{th} generation. By raising the number of local generations to 7, a schedule with almost an optimal makespan could be generated at the 7^{th} generation. However, with 11 generations, the performance goes down. This shows that cooperation among agents is necessary to enhance the efficiency of the generated schedule from a global perspective.

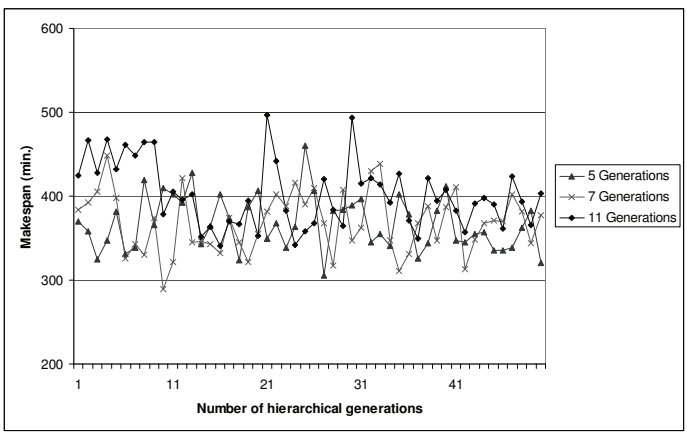


Figure 10.7: The effect of varying the number of local generations on the resulting makespan

10.3.2 Evaluation of Integrated Scheduling

In this section, the evaluation results of the framework under the integrated schedule generation mode are discussed. The evaluation aims at comparing the efficiency of the integrated scheduling compared to that of the production scheduling. The physical layout used for this experiment is captured in Figure 10.8, in which two FMCs are connected together through two work-in-progress buffers. Each FMC consists of a number of machines supporting one of the four tester types. The transportation of workpieces within an FMC is made available through two AGVs that travel with a fixed speed, amounting to 0.05 distance unit per second, in a predefined one-way path to which the machines are connected. Machines are connected to the transportation path through nodes that act as sensors guiding the movement of the AGV, as illustrated in Figure 10.8. These nodes are symmetrically distributed, by separating every two subsequent nodes with 10 units of distance.

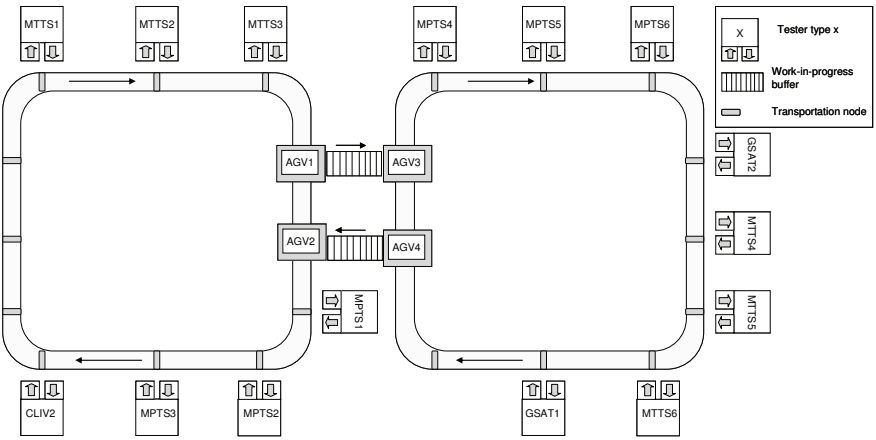
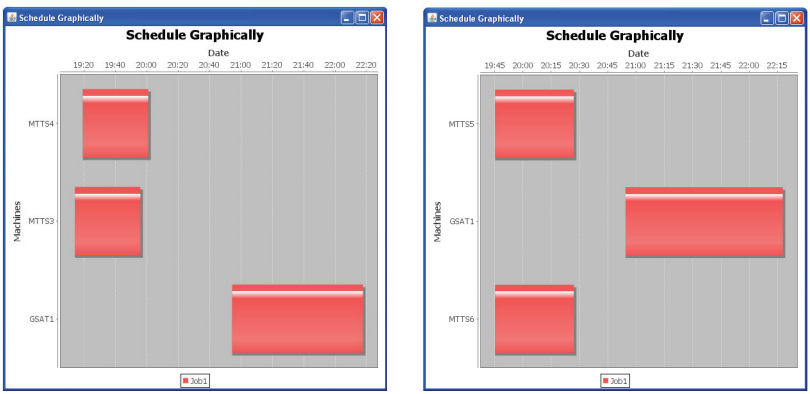


Figure 10.8: Physical layout used in testing the integrated schedule generation mode

A request for scheduling job1, corresponding to the manufacturing of 500 workpieces of part type4, is input to the framework. Before initiating this request, the machines are configured by setting up MTTS3 to part type4 and MTTS4 to part type5. The rest of the machines are not set up to any part. Initially, all machines are not allocated to any job. The scheduling process runs under the integrated scheduling mode twice: by setting the weight of the transportation costs W_T in (7.4) to zero and 0.75. While the former case corresponds to optimizing the schedule by ignoring the transportation costs, the latter case leads to incorporating the transportation costs into the schedule optimization with a relative weight of 75%, implying a weight of 25% to the machine costs.

Schedules resulting from the two cases are depicted in Figure 10.9. As shown in Table 10.1, the manufacturing of part type4 requires the processing of MTTS and GSAT test types. It is assumed that the manufacturing of these two operations is guided by the constraint that MTTS has to be processed before processing GSAT. The number of machines to be allocated for the incoming job request is governed by a percentage calculated by the heuristic, described in 7.5.1.2. Based on this heuristic, two MTTS machines and one GSAT machine are to be allocated for job1. In the first case, where the transportation costs are ignored, MTTS4 and MTTS3 are favored to other machines, since they are associated with the least setup costs and thus offer the earliest start time compared to other MTTS machines. For MTTS3 no setup time is required since it is already set up to the same part type and for MTTS4 only minor setup time is necessary because the machine is already set up for part type5, from the same part family. As a result, MTTS3 and MTTS4 are selected for the first operation. The selection of a GSAT machine follows the same principle and leads to randomly selecting GSAT1 since it offers the same proposal as GSAT2. In Figure 10.9(a), the resulting schedule is depicted, by plotting the associated transportation time required for moving workpieces between the MTTS machines and the GSAT machine.



a) Schedule resulting from setting $W_T = 0$

b) Schedule resulting from setting $W_T = 0.75$

Figure 10.9: Effect of integrated scheduling on the schedule efficiency

In the second case, where transportation costs are taken into consideration, MTTS5 and MTTS6 are favored to other MTTS machines, since they are located in the same FMC, where the GSAT machines are located. GSAT1 is selected since it is associated with lower transportation costs when moving workpieces from MTTS5 and MTTS6 along the one-way path depicted in Figure 10.8. The resulting schedule is shown in Figure 10.9(b). Although the schedule in Figure 10.9(a) yields higher machine setup, it leads to a lower makespan than the schedule in Figure 10.9(b). It was found that the resulting makespan is lowered by 31 minutes, amounting to around 17%

reduction of the makespan attained when ignoring transportation costs. However, this value depends on factors like the physical layout of the shop floor and the number and speed of the AGVs. In conclusion, the integration of the transportation scheduling into the production scheduling is a decision that can lead to enhancing the efficiency of the resulting schedule. Nevertheless, this decision has to be taken under consideration of the size and nature of the FMS.

10.4 Evaluation of Schedule Repair

In this section, the schedule repair mode is evaluated by investigating the efficiency and flexibility of the proposed disturbance handling method. While the efficiency is measured in terms of utility and stability of the resulting schedule, flexibility is measured in terms of reactivity. The utility of the applied actions was measured based on the added setup and the stability in terms of the total delay caused to the affected jobs on the resources applying the action. To visualize the schedule repair in reaction to a certain disturbance, the incoming of a higher priority job is considered. Before the incoming of Job3 (part type10), three MTTS machines are allocated to job1 (part type1) and job2 (part type10), as captured in Figure 10.10.

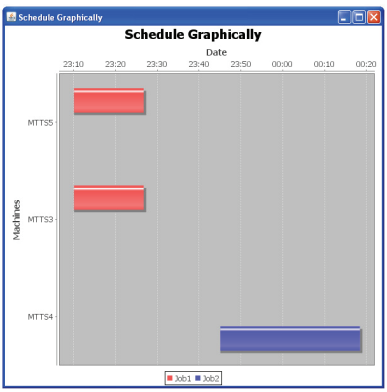
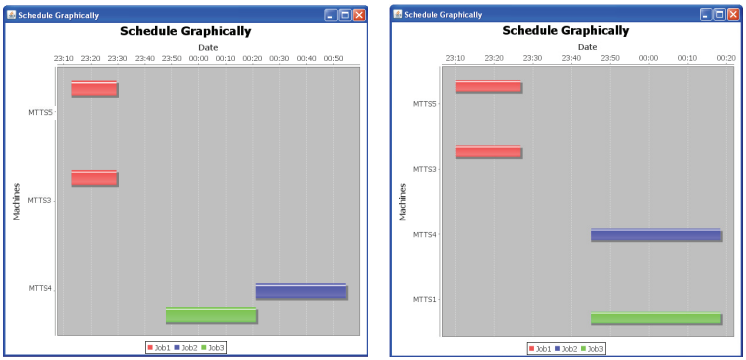


Figure 10.10: Schedule before the arrival of job3

For testing the efficiency of the generated schedule, two different runs were performed by setting the predefined weight w_u in (8.1) to 0.9 and 0.1, to emphasize the utility and the stability of the repaired schedule respectively. The schedule resulting from the former case is depicted in Figure 10.11(a) and the other one in Figure 10.11(b). When the predefined weight is configured to give a higher significance to utility at the cost of stability, machines yielding the lowest setup and thus the highest machine utilization are selected regardless of the affected jobs. This is manifested in selecting MTTS4 because it is already set up to the same part type of job3 and

avoiding the selection of another unallocated machine like MTTS1 to minimize the changeover of the machines (See Figure 10.11(a)). On the other hand, increasing the weight of stability at the cost of utility resulted in selecting MTTS1, which yields no effect on the already scheduled jobs but requires the changeover of machine MTTS1 (See Figure 10.11(b)). The configuration of this parameter is guided by the optimization objectives of the given FMS and is thus left to the user of the framework. A good compromise can be attained by setting the weight to 0.5 to result in a balanced consideration of both measures.



a) Schedule repair by favoring utility to stability

b) Schedule repair by favoring stability to utility

Figure 10.11: Schedules after the arrival of job3, using different utility and stability weights

Using this balanced configuration, several experiments have been conducted to test the reaction to the incoming of a higher priority job and the breakdown of a machine, as typical examples of external and internal disturbances. For testing external disturbances, the scheduled jobs were subject to a high degree of fluctuation with respect to priority. Jobs with high priorities corresponding to rush orders and representing 50% of the whole set were randomly inserted. On the other hand, two bottleneck machines allocated to about 40% of the existing jobs were subject to a breakdown with unknown duration causing forwarding jobs scheduled on the failed machine to peer machines. The schedule repair regarding the two scenarios was handled twice: without applying the optimization involved in the proposed disturbance handling method and with applying it. Table 10.4 illustrates the relative improvement of the makespan gained by the proposed approach in both cases. This improvement amounts to about 50%. The repair with optimization based on the proposed disturbance handling method was performed in less than 0.2 second on a 2 Duo 2.0 GHz processor, which shows good reactivity to the dynamic environment. Stability was measured in terms of the number of jobs that have to be updated in the old schedule. By considering the experiment of applying rush order disturbances, it was found that applying the optimization based on the expected impact on the affected jobs resulted in about 17% enhancement of the stability.

Table 10.4: Makespan of the repaired schedule with and without applying optimization

	Without optimization (min.)	With optimization (min.)
Machine breakdown	739.45	351.95
Rush order	687.88	376.42

10.5 Concluding Remarks

In light of the previous discussion and the presented experimental results, the proposed concept is evaluated with respect to the requirements. In what follows, the requirements, derived in section 2.3 are revisited and used to assess the proposed concept.

10.5.1 Efficiency of the Generated Schedules

Based on the experimental results, it has been shown that the generated schedules converge to the optimal solution. The flexibility available on the shop floor has been utilized to enhance the generated schedules. This is manifested in the allocation of several machines to an incoming job for the parallel execution of a given operation. This takes place by accounting for the costs incurred by the different resources to be allocated along with the expected performance out of the perspective of the concerned job. Test results have proved the support to maximizing the resource utilization under both the machine-centric and the comprehensive scopes. Consequently, the efficiency requirement is fulfilled by the proposed agent-based scheduling.

10.5.2 Flexibility to Short-Term Changes

The proposed concept exhibits flexibility to the dynamics of the environment by reacting efficiently to changes in part types and quantities. This leads to the contribution in the fulfillment of the process and volume flexibilities (See section 2.1.1.2). In addition, it has been shown that the proposed scheduling concept allows for a good balance of the system resources, which guarantees the availability of feasible alternatives for handling contingencies. This corresponds to the routing flexibility which together with the process and volume flexibilities constitutes the short-term flexibilities of an FMS, as discussed in section 2.1.1.2. Disturbances to the generated schedules are handled in a way that minimizes the deviation of the repaired schedule from the original schedule, which guarantees the required stability.

The structure of the agent-based scheduling allows for flexibly adapting the scheduling mode to the current environmental condition. This is manifested in adapting the scheduling technique according to whether the incoming requests represent the single or the multiple job case. Similarly, the agent-based communication and internal scheduling techniques are automatically adjusted based on whether the triggering event corresponds to a normal job request or a disturbance. Moreover, it has been shown that regardless of the scheduling mode, schedules are

generated or updated within a few seconds which results in good reactivity to changes in the scheduling environment. In summary, the requirement on short-term flexibility is satisfied.

10.5.3 Flexibility to Long-Term Changes

The proposed scheduling concept has been developed with special concern to the adaptation to long-term changes in the scheduling environment. As explained in section 2.1.1.2, long-term flexibility is a two-fold concept, consisting of product flexibility and expansion flexibility. It has been argued that the achievement of these two types of flexibility is to be reached through the configurability and the extensibility of the agent-based scheduling. The experience in modeling the IBM test line has shown the ease of adaptation of the realized framework to a specific FMS. No changes to the code was required neither to modeling the FMS in the framework nor to applying changes to it. As part of the evaluation of the integrated scheduling mode, the shop floor layout was changed to focus on just two FMCs, for illustration purposes. This adaptation to the shop floor layout was found to be straightforward. This shows that the extensibility of the FMS is fulfilled. Besides the extensibility, good support to the configurability has been demonstrated. Other than configuring the scheduling scope, optimization-related parameters can be easily configured to reflect the required scheduling strategy. For example, it has been shown that the relative significance of utility with respect to stability and that of machine utilization in relation to AGV utilization can be configured. The effect of this configuration has been illustrated based on the experimental results. By satisfying the extensibility and configurability, the proposed concept exhibits the required long-term flexibility.

10.5.4 Integration Ability in the Environment

The proposed agent-based scheduling provides good support to the integration ability into its environment represented in planning and control. In the course of testing the framework on the sample FMS, agents were connected to the simulated control, developed in plant simulation. Dynamic events, such as machine breakdown, were captured automatically by the concerned agents without affecting other agents. Likewise, generated schedules were delivered from agents to the concerned resources in the simulation model. On the other hand, the scheduling framework was found to provide interfacing channels to planning and provides insight into the internals of the scheduling process. This corresponds to the required transparency of scheduling, which is satisfied not only through the visualization of the final results but also through the provision of intermediate results. For example, in case of the simultaneous scheduling, the decision makers in the factory can get an insight into the schedules resulting from the different generations and not just from the final generation. This insight can aid in further planning and scheduling decisions.

In this chapter, the proposed concept has been evaluated based on the data of a real FMS, reported in the literature. Details about modeling the FMS in the agent-based framework have been given. Experimental results have been presented for the different scheduling modes supported by the proposed concept. The conducted experiments aimed at investigating the degree of fulfillment to the requirements, derived in section 2.3. It has been shown that the proposed concept offers a good coverage to the different targeted scheduling modes. Experimental results have shown that the proposed scheduling deals with the stochastic and dynamic nature of the scheduling environment flexibly and efficiently. Moreover, the integration ability of the proposed scheduling is attainable through the provided communication channels between the agents from one side and the simulated control and the interface to planning from the other side.

11 Conclusion and Future Work

In this work, a concept for the agent-based scheduling of FMS is proposed. In this chapter, the main characteristics and advantages of the proposed concept are highlighted. This is followed by pointing out the limitations of the concept. Finally, an insight into possible future work is given.

11.1 Summary and Contribution of the Research

The proposed concept caters for combining the efficiency of the generated schedule with the flexibility of the scheduling process. The concept presented in this thesis is characterized by the following features:

- The agent-based representation provides a comprehensive coverage of the requested jobs and the relevant resources. This feature along with the reactivity of agents allows for the dynamic capturing of events relating to the corresponding physical resource or the job. In this way, the reaction to the environmental dynamics is automated and the requirement on integration ability is satisfied.

- The scheduling system is abstracted from the details of the underlying FMS through the introduction of an environmental model. In this model, factory-specific attributes, which are relevant to scheduling, are captured. The separation of these attributes from the scheduling software results in facilitating the required extensibility which is part of the long-term flexibility.

- The scheduling system consists of agents representing jobs and resources, which correspond to entities from planning and shop floor control respectively. Resource agents are grouped based on their shared capabilities and controlled by service agents. Similarly, job agents are grouped based on their shared constraints and managed by job group agents. This results in a four-level heterarchical architecture. The agent-based representation together with the environmental model leads to the satisfaction of the long-term flexibility because of the openness of agents and their flexible communications.

- With their sensing and reacting abilities, agents capture their dynamic status and react accordingly. The autonomy and the decentralized decision-making of agents allow for good reactivity to short-term changes. The disturbance handling required for the short-term flexibility is achieved by developing a disturbance handling method and extending the proposed agent types with this method.

- Different scheduling modes are supported by the proposed concept, accounting for all the possible environmental configurations. The selection of the suitable scheduling mode occurs

either manually or automatically. The manual selection of the scheduling mode is intended to enable the decision makers of the FMS to adjust the scheduling process according to the adopted strategies and the scheduling objectives. On the other hand, the automatic selection of scheduling modes aim at making the scheduling process deal with disturbances and with different sizes of the requested jobs. This characteristic together with the previous one accounts for the requirement on short-term flexibility.

- Generated schedules are optimized based on simple problem-specific heuristics or genetic algorithms according to the size of the current problem space. The adoption of a genetic algorithm into the agent-based scheduling has been hindered by the inherently centralized structure of GA which does not suit the decentralized agent-based structure. This problem has been tackled by developing a concept for the hierarchical GA-based optimization. By employing a GA in agents involved in the search for schedules and allowing these agents to communicate together and combine and validate their achieved solutions, generated schedules are optimized from the global perspective with good reactivity.

- Schedules are optimized by taking the optimization objectives of the involved entities into consideration. This includes the maximization of the resource utilization and the minimization of the resulting makespan of the group of jobs. Accounting for the different optimization objectives takes place through the cooperation among the different concerned agents in the scheduling process. This cooperative optimization assumes either a machine-centric scope or a comprehensive scope. This feature along with the previous one leads to satisfying the efficiency requirement.

The proposed concept has been realized in a scheduling framework, which can be easily deployed for a given FMS. The framework has been used to conduct experiments on a model for a real FMS. The experiences in using the framework as well as the experimental results have proved the fulfillment of the concept to the entire set of requirements.

11.2 Advantages and Limitations of the Concept

The proposed concept represents a novel contribution to the scheduling of FMS. It was inspired by the results attained in the course of the state of the art survey, which revealed a considerable gap between theory and practice. The current research work targets bridging this gap by working out practice-oriented, theoretically well-established approach to replace the ad hoc methods, adopted in practice to complement the shortcomings of the current scheduling approaches. The strength of the proposed concept lies in its comprehensiveness in addressing the special requirements of FMSs and providing a balanced compromise between efficiency and flexibility.

It should be stressed that the proposed concept caters for the requirements of FMSs and is not intended for conventional manufacturing systems. It is concerned with scheduling and lays

special emphasis on its integrated nature with interrelated manufacturing functionalities, namely planning and shop floor control. However, it is not intended to solve problems falling in the realm of these functionalities. Examples of these problems include the tool allocation that is usually solved as a pre-release decision during planning. It provides support for integrating production scheduling with material handling in an attempt to enhance the quality of the generated schedule. Nevertheless, material handling is considered in this context with respect to the time taken to supply the required workpieces to the required machines. In other words, the objective is to take the transportation time into consideration and not to solve the material handling problem in its broader sense, including decisions like planning the supply of raw material. It is worth noting that despite these interrelated manufacturing decisions are regarded as exogenous to the proposed scheduling concept, the integration of the concept with agent-based solutions for these functionalities is envisioned as an area for future research with very good potential.

Another limitation that relates to the applied scheduling techniques corresponds to the optimization under the integrated scheduling mode. While production scheduling accommodates for both the sequential and the simultaneous modes, integrated scheduling is supported only under the sequential mode. This limitation lies in the applied algorithm and is dealt with by sequentially releasing requests in case they are to be fed into scheduling simultaneously.

11.3 Future Work

There are still open issues that remain to be addressed and are envisioned as useful areas of future work. An interesting research direction could be the development of a more sophisticated optimization algorithm to the integrated optimization. Currently, the optimization of the generated schedule occurs based on the iterative cooperation between operation agents and their corresponding resource agents. Operation agents undertake the validation of the schedules delivered from their resource agents and resulting from the local evolution inside these agents. Allowing resource agents, sharing the same operation, to communicate together can lead to speeding the convergence to a near-optimal solution. Similarly, the integrated optimization between transportation schedules and production schedules can be enhanced by developing a mechanism for exchanging proposals at the resource and the service levels before raising them up to the job level.

Another research direction corresponds to the extension of the agent-based scheduling with interrelated manufacturing functionalities like the tool allocation. In [BaGö09], a comprehensive agent-based model for incorporating tool allocation into the agent-based scheduling is proposed. By representing the shared tool magazines and the tool transporters as agents and allowing these agents to cooperate with the scheduling agents, the tool allocation process can be integrated with scheduling. In this way, the allocation of tools to machines can be dynamically organized based

on the communication among the concerned agents, which leads to improving the system flexibility.

Bibliography

[AbSv97] R. Abumaizar, J. Svestka: *Rescheduling job shops under random disruptions.* International Journal of Production Research 35:7, 2065-2082, 1997.

[Aydi07] M. E. Aydin: *Metaheuristic Agent Teams for Job Shop Scheduling Problems.* HoloMAS 2007, LNAI 4659, pp. 185–194, Springer-Verlag, 2007.

[Badr07] I. Badr: *Concept for the Development of Agent Oriented Embedded Real-Time Systems.* Master Thesis MT-2115, IAS, Universität Stuttgart, 2007.

[Badr08] I. Badr: *An Agent-Based Scheduling Framework for Flexible Manufacturing Systems.* International Journal of Computer, Information, and Systems Science, and Engineering (IJCISSE): Vol. 2 No. 2 Spring , pp.:123-129, 2008.

[BaGö09a] I. Badr, P. Göhner: *An Agent-Based Approach for Automating the Disturbance Handling for Flexible Manufacturing Systems.* In: Proceedings of the 14th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA09, 22.-26.09.2009, Mallorca, Spain, 2009.

[BaGö09b] I. Badr , P. Göhner: *An agent-Based Approach for Scheduling under Consideration of the Influencing Factors in FMS.* In: Proceedings of the 35th Annual Conference of the IEEE Industrial Electronics Society, (IECON-09), Porto, Portugal, 2009.

[BaGö10] I. Badr, P. Göhner: *Incorporating GA-Based Optimization into a Multi-Agent Architecture for FMS Scheduling.* In: Proceedings of the 10th IFAC Workshop on Intelligent Manufacturing Systems. Lisbon, Portugal, 2010.

[BaMi94] C. Basnet, J. H. Mize: *Scheduling and Control of Flexible Manufacturing Systems: A Critical Review.* International Journal of Computer Integrated Manufacturing, Vol. 7, No. 6, pp. 340-355, 1994.

[BMG08] I. Badr, H. Mubarak, P. Göhner: *Extending the MaSE Methodology for the Development of Embedded Real-Time Systems.* LADS2007, LNAI 5118, pp. 106 – 122, 2008.

[BSG10] I. Badr; F. Schmitt, P. Göhner: *Integrating Transportation Scheduling with Production Scheduling for FMS: An Agent-Based Approach.* In: Proceedings of the IEEE international symposium on industrial electronics (ISIE10), Bari, Italy, 2010.

[BuSc01] S. Bussmann, K. Schild: *An Agent-Based Approach to the Control of Flexible Production Systems.* In: Proceedings of the 8th IEEE Int. Conf. on Emergent Technologies and Factory Automation (ETFA2001), France, 2001, pp. 481-488 (Vol.2)

[BuSi01] S. Bussmann, J. Sieverding: *Holonic Control of an Engine Assembly Plant: an Industrial Evaluation.* In: Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference. Springer-Verlag: Berlin, Germany, 2001, pp. 169-174.

[CoJo02] P. Cowling, M. Johansson: *Using Real-Time Information for Effective Dynamic Scheduling.* European Journal of Operational Research 139(2002) 230-244.

[DBW91] D.M. Dilts, N.P. Boyd, H.H. Whorms: *The Evolution of Control Architectures for Automated Manufacturing Systems.* Journal of Manufacturing Systems, Vol. 10 (1991), pp. 79-93.

[Dies97] R. Diestel: *Graph Theory.* New York: Springer-Verlag, 1997.

[DuPe07] A. Duenas, D. Petrovic: *An Approach to Predictive-Reactive Scheduling of Parallel Machines Subject to Disruptions.* Annals of Operations Research. Vol. 159(1), 65-82, 2007.

[ElKh07] G. El Khayat: *Integral Approaches to Integrated Scheduling.* In: Multiprocessor Scheduling, Edited by Eugene Levener, ARS Press, Vienna, Austria / Pro Literatur Verlag, Mammendorf, Germany. ISBN: 978-3-902613-02-8, pp. 221-240, 2007.

[ELR06] G. El Khayat, A. Langevin, D. Riopel*: Integrated Production and Material Handling Scheduling using Mathematical Programming and Constraint Programming.* European Journal of Operational Research. Vol. 175,(3): 16 December 2006, pp. 1818-1832.

[FrGr96] S. Franklin, A. Grasser: *Is it an Agent or just a Program?* In: Proceedings of the 3rd conference about agent theories, architectures and languages, 1996.

[GeCh97] M. Gen, R. Cheng: *Genetic Algorithms and Engineering Design.* John Wiley & Sons, inc. 1997.

[Grav81] S. C. Graves: *A Review of Production Scheduling.* Operations Research, Vol.29, No.4, Operations Management. (Jul.-Aug., 1981), pp.646-675.

[Groo00] M. P. Groover: *Fundamentals of modern manufacturing.* John Wiley & Sons, New York, 2000

[Groo08] M. P. Groover: *Automation, Production Systems, and Computer-Integrated Manufacturing.* Peasron Education Inc., 2008.

[Gupt02] J. N. Gupta: *An Excursion in Scheduling Theory: An Overview of Scheduling Research in the Twentieth Century.* Production Planning & Control, Vol.13, No.2, 105-116, 2002.

[GuZh10] Q. Guo, M. Zhang: *An Agent-Oriented Approach to Resolve Scheduling Optimization in Intelligent Manufacturing.* Robotics and Computer-Integrated Manufacturing. Vol. 26, Issue 1, Feb. 2010, 39-45

[HaHo97] K-W Hansmann, M. Hoeck: *Production Control of a Flexible Manufacturing System in a Job Shop Environment.* International transactions in operational research, Vol. 4, No.5-6, 341-351(11), 1997.

[Herm06] J. W. Hermann: Rescheduling Strategies, Policies, and Methods: *Using the Rescheduling Framework to Improve Production Scheduling.* Handbook of Production Scheduling. Springer US, Vol. 89, 135-148, 2006.

[JAPS98] N. Jawahar, P. Aravindan, S. G. Ponnambalam, R. K. Suresh: *AGV Scheduling Integrated with Production in Flexible Manufacturing Systems.* Int. J. Adv. Manuf. Technol.14: 428-440, 1998.

[JASD06] J. Jerald, P. Asokan, R. Saravanan, A. Delphin Carolina Rania: *Simultaneous Scheduling of Parts and Automated Guided Vehicles in an FMS Environment using Adaptive Genetic Algorithm.* Int. Adv. Manuf. Technol. 29:584-589, 2006.

[Jenn01] N. R. Jennings: *An Agent-Based Approach for Building Complex Software Systems.* Communications of the ACM, 44 (4). pp. 35-41, 2001.

[JLMF07] S. Jacobi, E. Leon-Soto, C. Madrigal-Mora, K. Fischer. *MasDISPO: A Mutiagent Support System for Steel Production and Control.* AAAI 2007: 1707-1714.

[JoRa98] A. Jones, L. C. Rabelo: *Survey of Job Shop Scheduling Techniques,* NISTIR, National Institute of Standards and Technology, Gaithersburg, MD (1998).

[Kouv92] P. Kouvelis: *Design and Planning Problems in Flexible Manufacturing Systems: A Critical Review.* Journal of intelligent manufacturing (1992) 3, 75-99.

[KPM97] K. Kouiss, H. Pierreval, N. Mebarki: Using Multi-Agent Architecture in FMS for Dynamic Scheduling. Journal of Intelligent Manufacturing (1997) 8, 41-47.

[LaGö99] R. Lauber, P. Göhner: *Prozessautomatisierung I*, 3. Vollst. Überarb. Aufl. 1999, Springer-Verlag Berlin - Heidelberg - New York, 1999.

[Leit08] P. Leitao: *Agent-Based Distributed Manufacturing Control: A State-of-the-Art Survey.* Engineering Applications of Artificial Intelligence, 22, 979-991, 2008.

[LGP97] S. Lin, E. D. Goodman, W. F. Punch: *Investigating Parallel Genetic Algorithms on Job Shop Scheduling Problems.* In: Proceedings of the 6th International Conference on Evolutionary Programming VI. Vol. 1213, pp: 383 – 393, 1997.

[MaBi04] D. C. Mattfeld, C. Bierwirth: *An Efficient Genetic Algorithm for Job Shop Scheduling with Tardiness Objectives.* European Journal of Operational Research 155 (2004) 616–630.

[MaKo93] P. Maes, R. Kozierok: *Learning Interface Agents.* In: Proceedings of the 11th national conference on artificial intelligence, Washington, D. C., MIT Press, 1993.

[MoYa07] M. A. S. Monfared, J. B. Yang: *Design of Integrated Manufacturing Planning, Scheduling and Control Systems: A New Framework for Automation.* Int. J. Adv. Manuf. Technol. (2007) 33: 545–559

[Müll98] J. P. Müller: *Architectures and Applications of Intelligent Agents: A Survey.* The Knowledge Engineering Review, Vol. 13:4, 1998, 353-380

[OCP03] D. Ouelhadj, P.I. Cowling, S. Petrovic: *Utility and Stability Measures for agent-Based Dynamic Scheduling of Steel Continuous Casting.* In: Proceedings of the IEEE International Conference on Robotics and Automation, Taiwan (2003)

[Ouel03] D. Ouelhadj: *A Multi-Agent System for the Integrated Dynamic Scheduling of Steel Production.* Ph.D. Dissertation, The University of Nottingham, School of Computer Science and Information Technology, England (2003).

[OuPe08] D. Ouelhadj, S. Petrovic: *Overview on Dynamic Scheduling* (2008). Published online in the Journal of scheduling, 12 (4), 417-431, 2009.

[PRMP07] M. R. Paula, M. G. Ravetti, G. R. Mateus, P. M. Pardalos: *Solving Parallel Machines Scheduling Problems with Sequence-Dependent Setup Times using Variable Neighbourhood Search.* IMA Journal of Management Mathematics (2007) 18, 101–115.

[QHHW02] L. Qiu, W.-J Hsu, S.-Y.Huang, H. Wang: *Scheduling and Routing Algorithms for AGVs: A Survey*. International Journal of Production Research, Vol. 40, Issue 3, 2002.

[RaHo99] C. Rajendran, O. Holthaus: *A Comparative Study of Dispatching Rules in Dynamic Flow Shops and Job Shops.* European J. of Operational Research, 116 (1), 156-170 (1999).

[RaSt94] R. Rachamadugu, K. E. Stecke: *Classification and Review of FMS Scheduling Procedures.* Production Planning & Control, 1994. Vol.5, No.1, 2-20.

[RaTc90] J. Ranta, I. Tchijov: *Economics and Success Factors of Flexible Manufacturing Systems: The Conventional Explanation Revisited.* The International Journal of Flexible Manufacturing Systems, 2 (1990): 169-190.

[RFK04] R. Rangsaritratsameea, W. G. Ferrell Jr., M. B. Kurz: *Dynamic Rescheduling that Simultaneously Considers Efficiency and Stability.* Computers & Industrial Engineering 46 (2004) 1–15.

[RoDi00] A. Rossi, G. Dini: *Dynamic Scheduling of FMS using a Real-Time Genetic Algorithm*. Int. J. Prod. Res., 2000, Vol. 38(1):1-20.

[RuNo04] S. J. Russel, P. Norvig: *Artificial Intelligence – A Modern Approach.* Prentice Hall, 2004

[SaKi03] I. Sabuncuogluy, O. B. Kizilisiky: *Reactive Scheduling in a Dynamic and Stochastic FMS Environment.* Int. J. Prod. Res. Vol. 41(17), 4211–4231(2003).

[Schm09] F. Schmitt: *Conception and Realization of Agent-Based Transportation Scheduling for Manufacturing Systems.* Diplomarbeit Nr. 2257, IAS, Universität Stuttgart, 2009.

[SeSe90] A. Sethi, S. Sethi: *Flexibility in Manufacturing: A Survey.* The International Journal of Flexible Manufacturing Systems. Vol.2 no. 4 (1990): 289-328.

[Shen02a] W. Shen: *Genetic Algorithms in Agent-Based Manufacturing Scheduling Systems.* Integrated Computer-Aided Engineering, 9, 207-217, 2002.

[Shen02b] W. Shen: *Distributed Manufacturing Scheduling Using Intelligent Agents. IEEE Intelligent Systems*, 17(1): 88-94, 2002.

[Shiv06] H. K. Shivanand: *Flexible Manufacturing Systems.* New Age International (p) Limited, 2006.

[SMCS09] A. Serbencu; V. Mînzu; D. Cernega, A. Serbencu: *A Hybrid Metaheuristic for Solving Single Machine Scheduling Problem*. In: Proceedings of The 6th International Conference on Informatics in Control, Automation and Robotics. July, pp.68-74, 2009.

[Smit03] J. S. Smith: *Survey on the Use of Simulation for Manufacturing System Design and Operation.* Journal of Manufacturing Systems. Vol.22(2),2003, pp: 157-171.

[Stec83] K. E. Stecke: *Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems.* Management Science, Vol. 29, No. 3 (Mar., 1983), pp. 273-288.

[Stec85] K. E. Stecke: *Design, Planning, Scheduling, and Control Problems of Flexible Manufacturing Systems.* Annals of Operations Research. Vol. 3, No. 4 pp. 1-12, January, 1985.

[SuXu01] J. Sun, D. Xue: *A Dynamic Reactive Scheduling Mechanism for Responding to Changes of Production Orders and Manufacturing Resources.* Computers in Industry 46(2001) 189-207.

[SWH06] W. Shen, L. Wang, Q. Hao: *Agent-Based Distributed Manufacturing Process Planning and Scheduling: A State of the Art Survey.* IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, Vol.36, No. 4, July, 2006.

[TaHa88] J. Talavage, R. G. Hannam: *Flexible Manufacturing Systems in Practice: Applications, Design, and Simulation.* CRC Press, 1988.

[TeKu93] H. Tempelmeier, H. Kuhn: Flexible Manufacturing Systems: Decision Support for Design and Operation. Wiley-IEEE, 1993.

[THM09] M. T. Taghavifard, M. Heydar, S. S. Mousavi: *A Genetic Algorithm for Scheduling Flexible Manufacturing Cells.* Journal of Applied Sciences 9(1):97-104, 2009.

[Toko95] M. Tokoro: *An Agent is an Individual that has Consciousness.* In: Proceedings of the 2nd conference about agent theories, architectures and languages, 1995.

[Upto92] D. M. Upton: A flexible structure for computer-controlled manufacturing systems. Manufacturing Review, 1992. 5(1):58-74. http://www.people.hbs.edu/dupton/papers/organic/WorkingPaper.html

[VaMa00] J. Vancza, A. Markus: *An Agent Model for Incentive-Based Production Scheduling.* Computers in Industry 43 (2000) 173-187.

[VHL03] G. E. Vieira, J. W. Herrmann, E. Lin: *Rescheduling Manufacturing Systems: A Framework of Strategies, Policies, and Methods.* J.of Scheduling, 6 (1), Jan.-Feb.,2003.

[WaLi02] L. Wang, D. Li: *A Scheduling Algorithm for Flexible Flow Shop Problems.* In: Proceedings of the 4th world congress on intelligent control and automation. June10-14, 2002.

[Wall96] M. B. Wall: *A Genetic Algorithm for Resource-Constrained Scheduling.* Dissertation. Massachusetts Institute of Technology, 1996.

[Wall03] M. Wall: *Manufacturing Flexibility: What Constitutes the Holy Grail for Some is Just Business as Usual for Others.* Automotive industries, Oct, 2003.

[WeHy87] U. Wemmerlöv, N. Hyer: *Research Issues in Cellular Manufacturing.* Int. J. Prod. Res., 1987, Vol.25, No. 3,413-431.

[WGU03] T. Wagner, P. Göhner, P. Urbano: *Softwareagenten – Einführung und Überblick über eine alternative Art der Softwareentwicklung. Teil I: Agentenorientierte Softwareentwicklung.* In: atp – Automatisierungstechnische Praxis 45(2003), 10, 48-57.

[Whit06] Whitestein Technologies: *LS/TS API Documentation,* 2006.

[Wiki10] Wikipedia, the Free Encyclopedia. *Ford Model T*, http://en.wikipedia.org/wiki/Ford_Model_T, 2010.

[Witt90] R. J. Wittrock: *Scheduling Parallel Machines with Major and Minor Setup Times.* The International Journal of Flexible Manufacturing Systems, 2 (1990): 329-341.

[Wool01] M. Wooldridge: *An Introduction to Multi-Agent Systems.* John Wiley & Sons Ltd., 2001.

[WXZ08] S. Wang, L. Xi, B. Zhou: *FBS-Enhanced Agent-Based Dynamic Scheduling in FMS.* Engineering Applications of Artificial Intelligence, Vol. 21, No. 4, pp. 644-657, June, 2008.

[YaNa97] T. Yamada, R. Nakano*: Genetic Algorithms for Job-Shop Scheduling Problems.* In: Proceedings of Modern Heuristic for Decision Support, pp. 67-81, 1997, London.

Lebenslauf

Persönliche Daten:

11.06.1973	geboren in Kairo, Ägypten

Schulbildung:

1979 - 1985	Grundschule El Dwedar in Kairo
1985 - 1991	Gymnasium El Salam in Kairo - Abschluss “Thanawya Amma” (Abitur)

Studium:

1991 - 1996	Studium der Informatik an der Amerikanischen Universität in Kairo “American University in Cairo” Abschluss ”Bachelor of Science”

Weiterbildung:

1997 - 2000	Masterprogramm Informatik an der Universität Helwan in Helwan, Ägypten
	Abschluss M.Sc.

Berufstätigkeit:

1997 - 2001	Wissenschaftliche Mitarbeiterin an der Fakultät für Informatik, Universität Helwan in Helwan, Ägypten
Seit 2001	Wissenschaftliche Assistentin an der Fakultät für Naturwissenschaften, Universität Helwan in Helwan, Ägypten
2006 - 2010	Wissenschaftliche Mitarbeiterin am Institut für Automatisierungs- und Softwaretechnik der Universität Stuttgart