Robot Control Using UML and Multi-agent System

Vilem Srovnal

Department of Measurement and Control, VŠB – Technical University of Ostrava
17.listopadu15 , Ostrava, 70833, Czech Republic

and

Ales Pavliska

Department of Measurement and Control, VŠB – Technical University of Ostrava
17.listopadu15 , Ostrava, 70833, Czech Republic

ABSTRACT

Increased industrialization and new markets have led to an accumulation of used technical consumer goods, which results in greater exploitation of raw materials, energy and landfill sites. In order to reduce the use of natural resources conserve precious energy and limit the increase in waste volume. The application of disassembly techniques is the first step towards this prevention of waste. These techniques form a reliable and clean approach: "noble" or high-graded recycling. This paper presents a multi agent system for disassembly process, which is implemented in a computer-aided application for supervising of the disassembling system: the *Interactive Intelligent Interface for Disassembling System*. Unified modeling language diagrams are used for an internal and external definition of the disassembling system.

Keywords: Unified modeling language, Multi-agent system, Recycling, Disassembly planning, Supervision, Disassembly process

1. INTRODUCTION

Global competition and rapidly changing customer requirements are forcing major changes in the production styles and configuration of manufacturing organizations. Increasingly, traditional centralized and sequential manufacturing planning, scheduling, and control mechanisms are being found insufficiently flexible to respond to changing production styles and highly dynamic variations in product requirements. The traditional approaches limit the expandability and reconfiguration capabilities of the manufacturing systems. The traditional centralized hierarchical organization may also result in much of the system being shut down by a single point of failure, as well as plan fragility and increased response overheads.

Agent technology provides a natural way to overcome such problems, and to design and implement distributed intelligent manufacturing environments. It has been considered as an important approach for developing those industrial distributed systems. The agent technology could be applied to manufacturing enterprise integration, supply chain management, manufacturing planning, scheduling and control and materials handling [1],[4].

Unified modeling language-UML is a third-generation method, which provides system architects working on object analysis and design with one consistent language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. UML provides the basis for a common, stable, and expressive object-oriented development method. The UML is a standard accepted by the Object model group (OMG) and may also soon be adopted by the ISO group as well. The UML has sufficient expressive power to capture requirements and analysis models at a high level of abstraction and still be able to "drill down" into detailed design concepts such as threads, multitasking, resource control, etc.

This paper contains the work carried out to create the man-machine interface for supervision of disassembling system, where disassembly process is provided by multi agent system. The *Interactive Intelligent Interface for Disassembling System* (3IDS) composes of three main areas described in figure 1. Unified modeling language diagrams are used for an internal and external definition of 3IDS project. All UML diagrams are created in Rational Rose 2001 Evaluation version.

The base class for MAS is a ***Workshop***, from where is agents' environment built. Instances of classes ***Diary, Parts names*** and ***Parts information*** are independent entities that could be run from the both important objects *Analyze sequences* and *Disassembly process*.

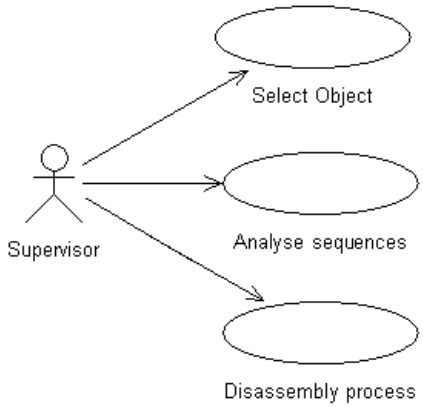

Figure 1: Use case diagram for main areas of 3IDS project

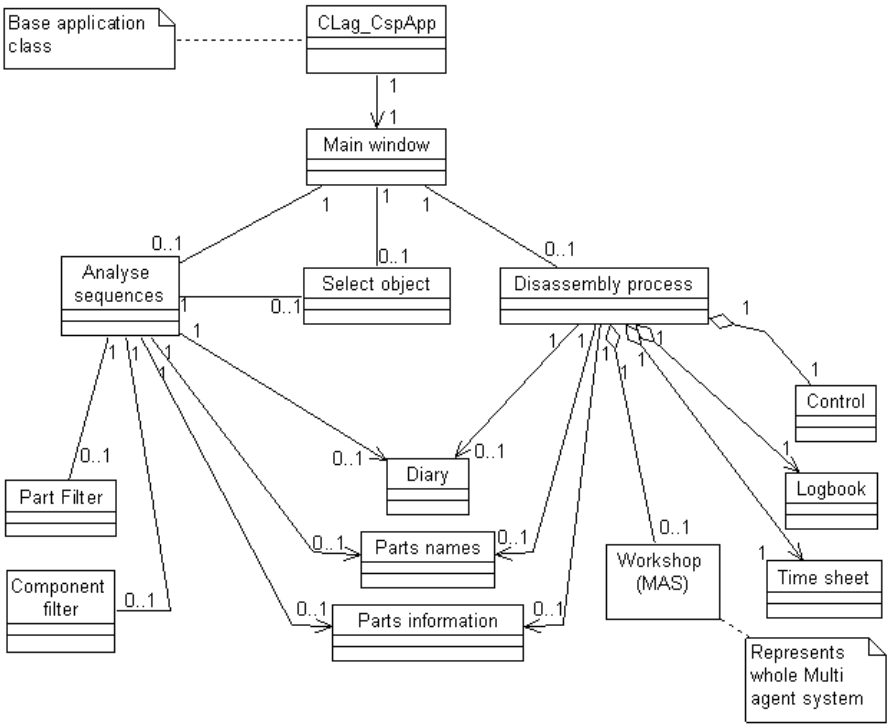

Figure 2: Class model of the 3IDS project

The supervisor has the possibility to select an object, to choose its sequence and to disassemble this object using choosing sequence.

Select Object represents *Select object* dialog. An object is an end-of-life product, for which exists demand on disassembly. There is a list with all possible objects.

Analyze sequences is very complex project part, which takes in plenty of tools for searching the best disassembling sequence for selected object. The output is an optimal dismantling sequence described by a numbered list of parts.

Disassembly process provides finally object disassembling. There are three selection modes:

- *Manual*, where the supervisor has to decide about all problems, those turn up after collision.
- *Semi automatic*, where an agents' architecture is used, but The supervisor agent isn't implemented.

Fully automatic, where agents' architecture is fully implemented.

Development Project Artifacts

The choice of what models and diagrams one creates has a profound influence upon how a problem is attacked and how a corresponding solution is shaped. Abstraction, the focus on relevant details while ignoring others, is a key to learning and communicating. Because of this:

• Every complex system is best approached through a small set of nearly independent views of a model. No single view is sufficient.

• Every model may be expressed at different levels of fidelity.

• The best models are connected to reality.

In terms of the views of a model, the UML defines the notation and semantics for the following domains:

The Use Case Model - describes the boundary and interaction between the system and users. Corresponds in some respects to a requirement model.

The Class (Object) Model - describes the classes and objects that will make up the system.

The Dynamic Models - contains four diagrams: Sequence diagram, State chart, Activity diagram and Process diagram.

The Collaboration Model - describes how objects in the system will interact with each other to get work done.

The Physical Models - A Component Model describes the software (and sometimes hardware components) that make up the system. A Deployment Model describes the physical architecture and the deployment of components on that hardware architecture.

Project class model

The 3IDS project includes 15 dialogs and 36 classes. The main class structure is on figure 2. There are shown relations between classes, types of their collaboration and numbers of their instances. Multi agent system is covered up without better specification because of its complexity. In figure 2, all classes (except CLag_CspApp) have imaginative names to be simpler to understand their missions. This routine is used in all next sections.

A ***CLag_CspApp*** class is an absolute base class, so-called *Application class, which exists* in every Windows application created by MFC. Each *Application class* contains routines and methods that are necessary for creating an application. Nominally, it contains the main application constructor and destructor, the framework-calling methods and its IID.

A ***Main Window*** class represents a modal *Main dialog,* its instance is created in the *CLag_CspApp* and it exists as long as

the 3IDS project is running. Always only one of objects (instances of classes): *Analyze sequences, Select object* and *Disassembly process* could be created from *Main Window,* because these objects contain modal dialogs. Therefore, it's impossible to create another object from the *Main Window* before the existing object is deleted (its destructor is called).

An ***Analyze sequences*** class contains a modal *Analyze* dialog. After possible sequences for the selected end-of-life product are loaded, the *Analyze* dialog is created to enable choosing the right dismantling sequence. From the *Analyze sequences* class could be created always only one instance of a *Part filter* class, a *Component filter* class or a *Select Object* class. In addition, it's possible to create instances of classes *Diary*, *Parts names* and *Parts information* that are represented by modeless dialogs.

A ***Select object*** class contains an algorithm for all possible sequence generating from the contact-relation model and the economical specifications of a product, and a modal dialog enabling selection of an end-of-life product.

A ***Disassembly process*** class contains modal dialog and data structures for disassembly. Immediately after creating of the *Disassembly process* dialog, instances of Control*, Logbook* and *Time sheet* classes are created. In addition, it's possible to create instances of classes *Diary*, *Parts names* and *Parts information* that are represented by modeless dialogs. When a disassembly process is running and a collision is detected, an object of *Decision* class or *Decision (MAS)* class could be created in dependency on MAS using. These classes are represented by modal dialogs that are not implemented in hierarchical tree in figure 2. If the disassembly process is started and Multi agent system (MAS) is used, all agents have to be created.

Instances of ***Control, Logbook*** and ***Time sheet*** classes exist as long as the instances of Disassembly process class. Process output data are sent every simulation second to the *Logbook* and to the *Time sheet* to be displayed. Abstractedly from them, the Disassembly process object to catch all changes on the Supervisor's control panel continually monitors the instance of the Control class.

The ***Part filter*** class and the ***Component filter*** class harbor algorithms for finding such sequences that correspond to selected conditions in their dialog windows. Corresponding sequences are subsequently marked in the *Analyze sequences* class dialog.

2. The Interactive Intelligent Interface for Disassembling System

The 3IDS is a project for supervising of disassembly process. The 3IDS project enables to select an end-of-life product, to analyze its sequences, to choose the best one, to control and to monitor a disassembly process using multi agent system.

The 3IDS project represents a section of the disassembling system. Whole disassembling system could be divided into three main sections: the CAD model, the Supervising cell (3IDS) and the Disassembling cell. The CAD model represents a difficult step in a disassembly planning system. The CAD models are used for a fully automated analysis of contacts among the parts evaluated along the axes XYZ of the CAD reference system. Data, together with the description of the connections, are transferred to a numerical contact-relation model to be easier handled by the second section of the disassembling system. The contact-relation model expresses the types of contact-relationships between parts of the product [3]. These models are input of the 3IDS project. Disassembling cell represents the core of disassembling system. It contains robots with specialized tools and operators.

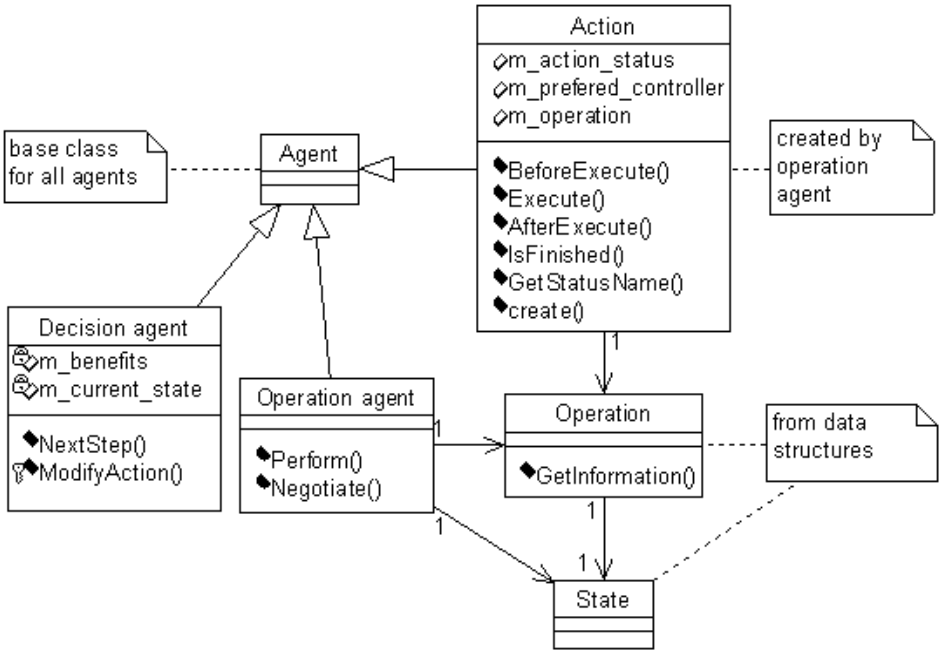

Figure 3: Class structure of agents in the decision-making unit

In this case, system is designed for using of one operator and one robot. The 3IDS project composes of two main areas: *Analyze sequences* and *Disassembly process*. The first one is very complex project part, which takes in plenty of tools for searching the best disassembling sequence for selected end-of-life product. The output is an optimal dismantling sequence described by a numbered list of parts. The second one provides to simulate the final product disassembly. It works in three modes: *Manual mode*, where MAS is not used, and two *Automatic* modes that use MAS for disassembling.

3. MULTI AGENT SYSTEM REALIZATION

In the 3IDS project, the multi agent system (MAS) is designed to work based on a predicted scenario. It's designed to deal with decision making in a disassembly process by choosing the best disassembling sequence, when a non-expected collision is occurred. The disassembling sequence is an ordered sequence of product parts to be disassembled.

The MAS is realized in two different modes: the *Semi-automatic* mode and the *Fully- automatic* mode. If a collision is occurred, the MAS will try to replace the current disassembling sequence from its last disassembled part to its end with another fraction. A fraction is a part of whole sequence. If the MAS finds more than one possible fraction, it will choose the fraction which first part best corresponds to adjusted criteria. The supervisor sets these criteria in the main menu of the *Disassembly process* window. There are five different criterion types: the disassembly cost, the part value, the benefit price, the part weight and the disassembling time estimation. For each type, the MAS could look for the highest value or the lowest value in dependence on the supervisor's setting. If the *Semi-automatic* mode is set, it asks supervisor for help how to continue. If the *Fully-automatic* mode is set, the supervisor agent is implemented and he calls the operator to try to continue on disassembly of the current part at the place, where the robot ended with a collision. However, if the operator doesn't succeed, the supervisor agent has to cancel current disassembly process.

General structure

The general structure of the MAS consists of four logical units' [2]:

Data structures are the set of structures, which contains input and preprocessed data. System receives the set of possible sequences to disassembly and their economical specifications.

Decision making is the set of agents, which has references to data structures and which dynamically choose the best step from the set of sequences to perform some operations depending on the end-of-life product state and the current situation. The *Decision unit* has to determine which operation is the best. If some operation fails, it has to decide what another operation could be performed or that the intervention of the *operator* or the *supervisor* is needed.

Controllers are the set of agents that can interact with a robot or an operator. The Controller agents execute the operations, which was chosen by the *Decision making unit*. They are used for executing of an action of *Robot* and *Operator* objects.

Supervisor is a person who interacts with the system and controls it. To perform some actions on the product he can use his agent (*Supervisor agent*) to "tell" him execute some actions.

Decision making unit

There are three types of agents in the *decision-making unit* (fig.3):

- *Decision*
- *Operation*
- *Action*

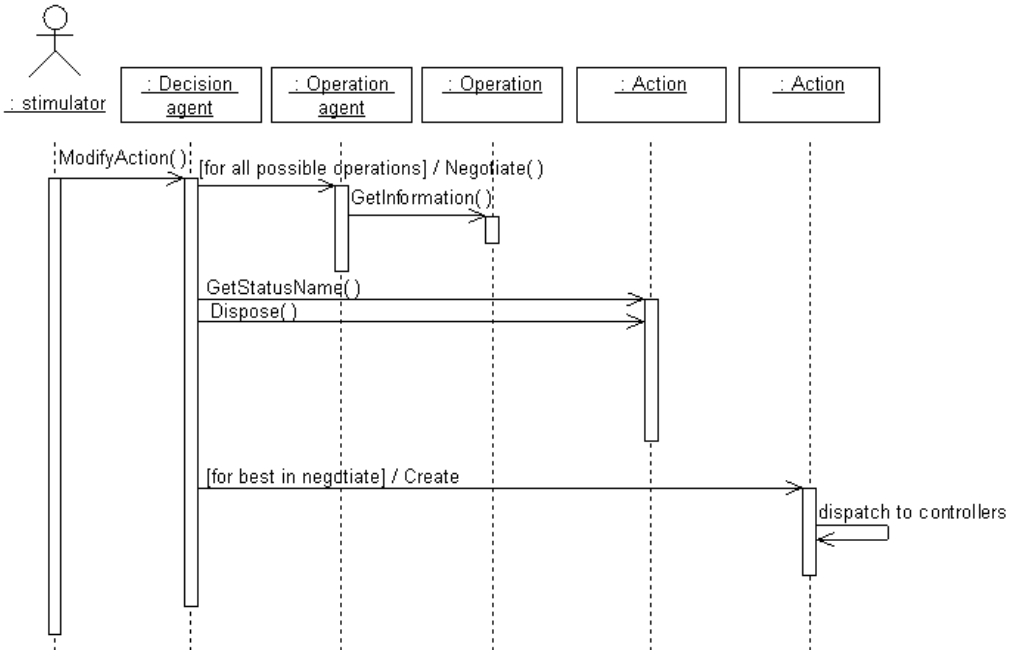

Figure 4: Sequence diagram for action modifying

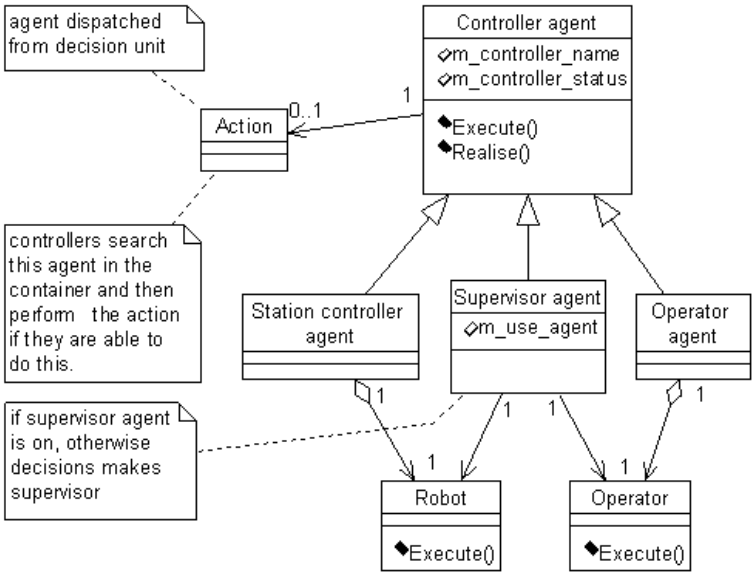

Figure 5: Class structure of controllers

Decision agent is one agent, which makes decision basing on the information from *operation agents*. This agent has its current *state*. On the beginning, its state is the selected dismantling sequence. Every step the agent updates its current *state*. On a collision in a disassembly process, this agent creates *operation agents* for all possible *operations*. To decide which *operation* is the best to perform on the current *state,* the d*ecision agent* asks the *operation agents* (calling *Negotiate* method).

The d*ecision agent* chooses the *operation agent,* which returns the best-fit value.

Operation agent is the agent, which is created for each possible *operation*. This agent has a method *Negotiate,* which returns a numerical value of the economical specification of the first potential *operation* in the state, which could be performed by the *operation agent*.

Action agent is the agent, which is created after making a decision. It is created to perform a chosen *operation*. It is a mobile agent and it can be dispatched between two units: the *decision making unit* and the *controllers unit*.

If the *controller* has a problem with executing the action, the *action agent* is dispatched back to the *decision unit*. In this case, another decision has to be made or the *action agent* has to be canceled (fig.4).

Controllers unit

There are three agents in *controllers unit* (fig. 5):

- *Station controller agent*
- *Supervisor agent*
- *Operator agent*

All those agents are extended from its base class: the *controller agent*. There are also two objects in this unit: the *Robot* and the *Operator*. The agents from the controllers unit control those objects.

Each *controller agent* is stimulated by the system. Each free (not busy) *controller agent* (rather its specialization) searches for an *action agent* in the *agents' container*. If the *action agent* exists and waits for its realization and if the *controller agent* is able to execute this action (fig. 6), then the *controller agent* tries to execute it. If the action was successfully executed, the *action agent* is disposed. Then the *decision unit* will create the next *action agent,* which will represent another *operation.* If execution was failed, the *action agent* is not disposed. It is dispatched back to the *decision unit* and it is there processed (it is analyzed, disposed and the new *action agent* is created if necessary).

4. CONCLUSION

This paper describes a recycling platform, which implements the "noble recycling" concept for technical end-of-life products. It concentrates on the description of multi agent system, which is used in the 3IDS project to increase the reactivity of disassembly process. All project algorithms are implemented using MS Visual C++ 6.00 basing on Unified Modeling Language (UML) model. There are few elements of the system, which might be further worked out. The most important element of further work will be the connection with real or simulated disassembling cell. Simulation could be effected in Robcad environment from Technomatix. Within the "*Rester propre"* project, research works proceed just in Robcad environment to simulate whole disassembling cell containing a robot, an operator and an end-of-life product. (The *Rester propre* project has originated on *Institut National Polytechnique de Grenoble* in *Laboratoire d'Automatique de Grenoble.* It deals with supervision and control systems in high-graded recycling.) For materialize, the agents' environment upgrading is needed. *Robot* object is currently an abstract object, which simulates execution of some actions. In the future, this object has to be replaced by a real robot object, which will translate all information from the *action* agent to a language of the robot and passes it to this robot.

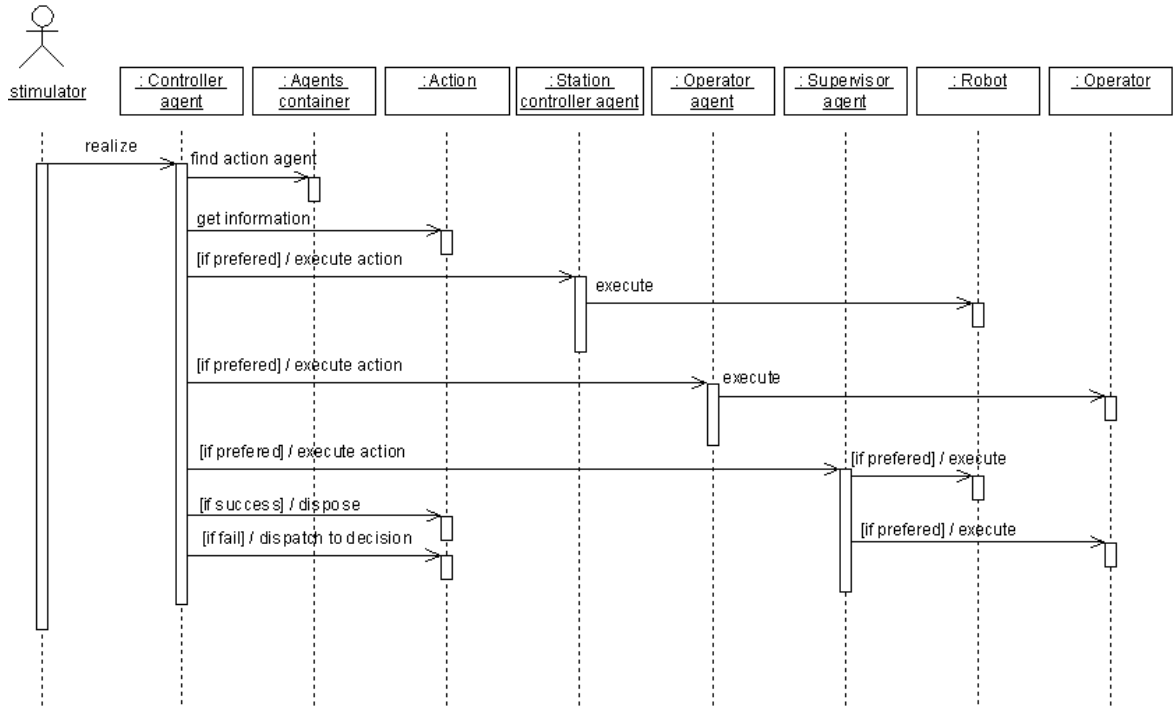

Figure 6: Sequence diagram for action execution

The goal of this work was to develop a computer-aided application for disassembling system supervision to provide these postulates: Generation of disassembling sequences from the product contact-relation model in accordance to fixed criteria. A graphical confrontation of disassembling sequences to enable choosing the best one, a simulation of the product disassembling while using selected dismantling sequence. A Multi-agent system using for disassembly process.

All these postulates were fully accomplished by creating of an *Interactive Intelligent Interface for Disassembling System* (3IDS). Moreover, 3IDS application offers a possibility of simple end-of-life product changing, a numerical confrontation of disassembling sequences by sequence filtering, and disassembly process supervision in the mode that doesn't support multi agent system using, where all decisions depend on supervisor's judgement.

In spite of that, there are few elements of the system, which might be further worked out. The most important element of further work will be the connection with real or simulated disassembling cell. *Disassembly process* window of 3IDS application is designed to an easy adding of new components such as service cameras or real-time chat between supervisor and operator to facilitate a graphical connection with disassembling cell. For materialize, the agents' environment upgrading is needed. *Robot* object is currently an abstract object, which simulates execution of some actions. In the future, this object has to be replaced by a real robot object, which will translate all information from the *action* agent to a language of the robot and passes it to this robot.

Acknowledgement

The 3IDS project was finished on VSB Technical University of Ostrava, Czech Republic. The Ministry of Education of Czech Republic supplied the results of the project CEZ: J17/98:272400013 with subvention.

The Grant Agency of Czech Republic supplied the results of the project 102/01/0803 with subvention.

5. REFERENCES

[1] K.Kwiatkowski, A.Skaf, Decision making system for disassembling process, Laboratoire d'Automatique de Grenoble , INPG, France, 2000

[2] G.Dini, F.Failli, M.Santochi, A disassembly planning software system for the optimization of recycling process, Department of Production Engineering, University of Pisa, Italia, 1999

[3] D.Chevron, S.Drevon, Z.Binder, Design and implementation of a man-machine interface for disassembling cell, Laboratoire d'Automatique de Grenoble, INPG, Grenoble, France, 1997

[4] K.Cetnarowicz, J.Kozlak, Multi-Agent System for Flexible Manufacturing Systems Management, In: Pre-Proceeding International Workshop CEEMAS'01. AGH Krakow 2001, ISBN 83-915953-0-7, pp.41-50