A MULTI-AGENT APPROACH FOR DESIGN CONSISTENCY CHECKING

A dissertation
submitted to the
Department of Computer & Information Sciences,
De Montfort University
in partial fulfilment of the requirements for the degree of
Doctor of Philosophy
by

Victor Taratoukhine

BRITISH LIBRARY

IMAGING SERVICES NORTH
Boston Spa, Wetherby
West Yorkshire, LS23 7BQ
www.bl.uk

APPENDIX F

NOT DIGITISED BY REQUEST OF THE UNIVERSITY

ABSTRACT

The last decade has seen an explosion of interest to advanced product development methods, such as Computer Integrated Manufacture, Extended Enterprise and Concurrent Engineering. As a result of the globalization and future distribution of design and manufacturing facilities, the cooperation amongst partners is becoming more challenging due to the fact that the design process tends to be sequential and requires communication networks for planning design activities and/or a great deal of travel to/from designers' workplaces. In a virtual environment, teams of designers work together and use the Internet/Intranet for communication. The design is a multi-disciplinary task that involves several stages. These stages include input data analysis, conceptual design, basic structural design, detail design, production design, manufacturing processes analysis, and documentation. As a result, the virtual team, normally, is very changeable in terms of designers' participation. Moreover, the environment itself changes over time. This leads to a potential increase in the number of design. A methodology of Intelligent Distributed Mismatch Control (IDMC) is proposed to alleviate some of the related difficulties.

This thesis looks at the Intelligent Distributed Mismatch Control, in the context of the European Aerospace Industry, and suggests a methodology for a conceptual framework based on a multi-agent architecture. This multi-agent architecture is a kernel of an Intelligent Distributed Mismatch Control System (IDMCS) that aims at ensuring that the overall design is consistent and acceptable to all participating partners.

A Methodology of Intelligent Distributed Mismatch Control is introduced and successfully implemented to detect design mismatches in complex design environments.

A description of the research models and methods for intelligent mismatch control, a taxonomy of design mismatches, and an investigation into potential applications, such as aerospace design, are presented. The Multi-agent framework for mismatch control is developed and described. Based on the methodology used for the IDMC application, a formal framework for a multi-agent system is developed.

The Methods and Principles are trialed out using an Aerospace Distributed Design application, namely the design of an A340 wing box. The ontology of knowledge for agent-based Intelligent Distributed Mismatch Control System is introduced, as well as the distributed collaborative environment for consortium based projects.

Keywords: Engineering Design, Multi-agent systems, Mismatch Control, Aerospace Design.

STATEMENT OF CONTRIBUTION

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which, to a substantial extent, has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

In this thesis the new Intelligent Distributed Mismatch Control (IDMC) approach was introduced and described. The results described in this thesis are an original contribution to the problem at hand. Novel aspects of the contribution include: a new taxonomy of design mismatches; a new conceptual multi-agent framework; and the development of an Intelligent Distributed Mismatch Control System (IDMCS).

Various aspects of the IDMC approach have been described in papers published elsewhere. The approach was first outlined at the IEEE Information Visualization Confererence, London, 1999. The work described in this thesis has been published in [Bechkoum and Taratoukhine, 1999a], [Taratoukhine and Bechkoum, 1999a], [Taratoukhine and Bechkoum, 2000a], [Taratoukhine, Bechkoum, Stacey, 2001a], and several other publications.

The work is original and all sources of information have been acknowledged.

Victor Taratoukhine

ACKNOWLEDGEMENTS

Although all other work in the thesis is entirely my own responsibility, I would have been unable to complete it without the help of my supervisors and colleagues. In particular I would like to thank Dr Kamal Bechkoum, Dr Martin Stacey, Dr Gordon Clapworthy, and Dr Claudia Eckert.

I would also like to thank our British and Russian aerospace partners for their assistance in setting up and running the experiment described in Chapter 8.

Finally, I would like to thank my colleagues from Computer and Information Sciences Research Lab for a supportive research environment and useful suggestions.

Victor Taratoukhine

CONTENTS

ABREVIATION

DAI - Distributed Artificial Intelligence
CAD - Computer Aided Design
CAM - Computer Aided Manufacturing
CAE - Computer Aided Engineering
PDM - Product Data Management
CE - Concurrent Engineering
CSCW- Computer Supported Collaborative Work
IDMC - Intelligent Distributed Mismatch Control
IDMCS - Intelligent Distributed Mismatch Control System
MAS - Multi-agent system
M_1 - model of system
M_{str} - set of elements of structure M_1
M_{par} - set of elements of parameters M_1
M_{par}^{cr} - critical parameters
M_2 - distributed knowledge-base
W_i - designer's world
T - time of design
Tax - taxonomy
DA - design agent (D-agent)
CA - control Agent (C-agent)
FB - facts base
KB - knowledge-base
GI - global interface
LI - local interface
K - corrector
I - inference engine
L_I - information language
L_c - control language
L_c^c - control language for C-agents interrelations
CP - communication protocol

LIST OF ILLUSTRATIONS

LIST OF TABLES

This thesis is dedicated to my family

Chapter 1 Introduction

1.1 Problem Definition and Motivation

Research into the use of knowledge engineering in design has become widely accepted as a fast growing subfield of Artificial Intelligence (AI). Increasing numbers of researchers, and research groups, are active within this emerging subfield. From advocates of "knowledge intensive" CAD/CAM/CAE [Tomiyama, 1993; Tomiyama et al, 1994] to promoters of broader "intelligent CAD frameworks" (e.g. [Akman et.al. 1989, 1990; Bento & Feijo, 1997; Bento, 1998, 2000) the common thread is the use of AI tools and techniques to provide automatic and semi-automatic solutions to the problem. These solutions aim at increasing the "intelligence" of existing CAD/CAM/CAE systems [Brown and Grecu, 1997; Brown and Grecu, 1996; McMahon & Browne, 1993; Rzevski, 1998; Oliveira & Bento, 1996; Frost and Cutkosky, 1996].

The AI technologies used are varied and include expert systems [Bechkoum, 1997; Bechkoum & Taratoukhine, 1999a, 1999b; Katragadda, 1997; Knowledge Based Engineering at Airbus, 1998], genetic algorithms and evolution programming [Gero, 1998], fuzzy logic [Gero, 1998; Semoushin at al., 1997] and multi-agent systems [Bento & Feijo, 1997; Sycara, 1998; Dunskus et. al, 1995; Cutkosky at al., 1993]. Hybrid methods combining more than one technology have also been used [Taratoukhine at al., 1997; Belov, 1989; Joslyn at al., 1999].

It is fair to say though, that design engineers are still sceptical about the ability (or inability) of current intelligent design-support systems. For example, even when endowed with some sort of intelligent behaviour, existing CAD/CAE systems cannot handle several types of inconsistencies that may occur during the design phase. For instance, according to Akman, Hagen and Tomiyama [Akman, Hagen and Tomiyama, 1990], "Current CAD systems are not fully able to recognise inconsistencies in their input data. To exacerbate the situation, the final output of conventional systems can be so impressive that many errors go unnoticed for they exceed the mental capacity of designers".

Mainly due to the complexity of the design process, existing solutions tend to approach the problem from a very specific angle. For example, commercial systems such as CATIA [CATIA] and I-DEAS [IDEAS] do provide facilities for assembly mismatch control, but their approaches are more focussed on the tolerances. Other contributions [Bechkoum, 1997] are constrained by the number and types of mismatches considered. Often, attention is given to a few geometric mismatches only, with very little concern about (say) material or cost considerations.

Moreover, even when the proposed approach is successful in detecting a design anomaly rarely does it suggest a satisfactory way to resolve the problem. In most of the previous work all the design knowledge is centralised into one unit: the knowledge base [Akman et. al, 1990; Bechkoum, 1997; Gero, 1998]. The centralisation of knowledge coupled with the absence of a negotiation mechanism (between all parties involved in the design) makes the process of predicting the impact of any modification an (almost) impossible task.

We re-inforce here the view that a multi-agent approach can tackle many of the problems posed by the centralisation of knowledge into a single Knowledge Base.

The use of intelligent agents as independent distributed knowledge entities promises to provide the missing link. In this context, the investigation of methods and principles of organisation of multi-agent systems for mismatch design is investigated. This multi-agent architecture is at the heart of an intelligent distributed mismatch control system (IDMCS) that aims at ensuring that the overall design is consistent and acceptable to all. Of course, the number of different questions should be investigated such as the design mismatches can be detected earlier as the result design process is cheaper, mismatches can be resolved faster, mismatches can be avoided, etc. The next section reviews EDID project. This project was a starting point of the research in design consistency checking.

1.2 EDID-IMCS Project

In [Bechkoum, 1997] K. Bechkoum describes an Intelligent Mismatch Control System (IMCS) which has the potential to detect some types of mismatches as part of the EDID Project (Environment for Distributed Integrated Design).

Main objectives of EDID project

EDID project addresses the field of Distributed Collaborative Design in the European Aerospace Industry sector. Its major goal is to prepare for new ways of working that should increase both productivity and quality in multi-partner, space design projects. The new processes envisioned to encompass CSCW techniques including multimedia communication capabilities. These capabilities, given the work locations geographically dispersed over Europe, necessitate a broadband trans-European communication system featured with powerful and flexible services that the project will identify.

Technical Approach

The experiment is based on a scenario for a simplified representation of a satellite design. The satellite is split into parts designed by different partners. The core of the scenario addresses the negotiation process between prime contractor and contributing partners at different stages of the design. One example is overall design consistency verification, where part designs are integrated and mismatches have to be detected and solved by actors geographically dispersed but collaborating through a technical conferencing environment.

The IMCS implementation is an important step towards a more comprehensive solution but is far from being defects-free. For example, the number and types of mismatches handled by the system is narrowed down to a few geometric mismatches. Also, the system detects mismatches, but rarely suggests a way to resolve them.

Intelligent Distributed Mismatch Control (IDMC) is advanced development of IMCS is outlined in this thesis. The next sub-section reviews the main differences between EDID-IMCS and IDMC.

1.3 IDMC Project

The work presented here takes the IMCS' development one step forward. A new multi-agent architecture is proposed which gives the IMCS the ability to handle issues peculiar to the nature of distributed design.

This multi-agent architecture is at the heart of an intelligent distributed mismatch control system (IDMCS) that aims at ensuring that the overall design is consistent and acceptable to all. In the Fig. 1 the overview of IDMCS Project and EDID Project is presented.

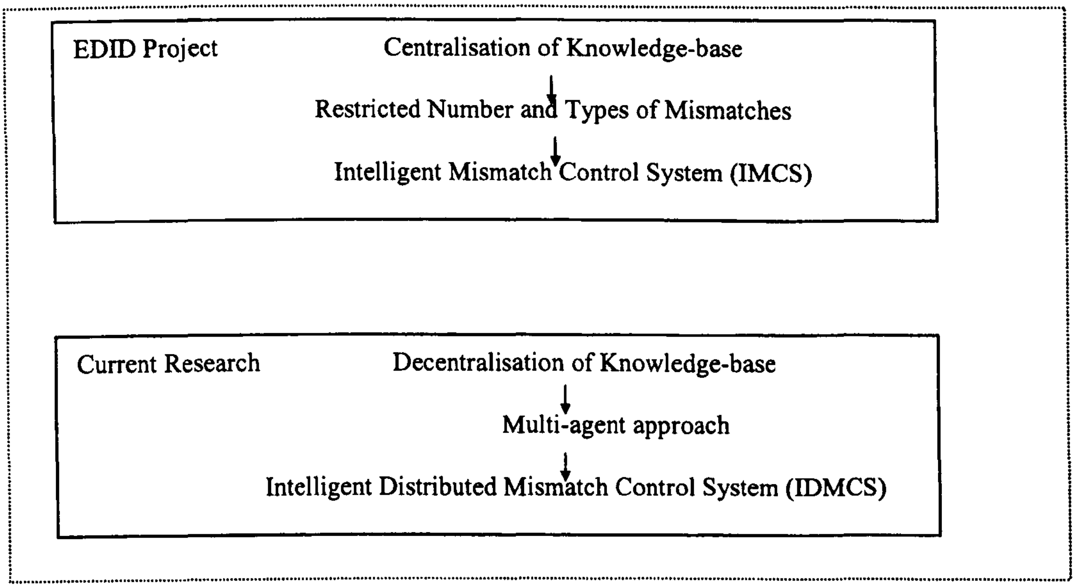

EDID Project
Centralisation of Knowledge-base
Restricted Number and Types of Mismatches
Intelligent Mismatch Control System (IMCS)
Current Research
Decentralisation of Knowledge-base
Multi-agent approach
Intelligent Distributed Mismatch Control System (IDMCS)

Figure 1: EDID and IDMC Projects

De Montfort University initially supported the IDMC project as PhD three years project and later received an industrial support (in kind) from Aviation Euro-Russian Consortium, TUPOLEV Corp and AVIASTAR. Also, participation of Electroimpact Inc (USA) was very helpful during visit AIRBUS/BAE Systems facility in Chester, UK.

1.4 Aims and Research Methods

This section outlines the aims, methods and outcomes of this research. Also the general plan of investigation is presented (Fig. 2).

Research Problem:

For many years the design and manufacture of major European complex products, such as satellites, airplanes and cars has been distributed across the continent. As the result of globalization and future distribution of design and manufacturing facilities, the cooperation amongst partners is more challenging. The design process tends to be sequential and requires centralised planning teams and/or a great deal of travel to/from distributed designers. In a virtual team, designers work together and use a Internet/Intranet for communication. The design is a multi-disciplinary task that involves several stages. These stages include input data analysis, conceptual design, basic structural design, detail design, production design, manufacturing processes analysis, and documentation. As a result, the virtual team, normally, is very changeable in terms of designer's participation. Moreover, the environment itself changes over time. This leads to a potential increase in the number of design mismatches. A methodology of Intelligent Distributed Mismatch Control is needed to alleviate some of these problems.

The research aim:

The main aim of this research is to develop a methodology, models and tools for detecting design inconsistencies in a distributed design environment.

Research question:

In our view the definition of research question should motivate the researcher during the project. Research question adopted in this thesis is based on the hypothesis that distributed AI, particularly multi-agent systems can be a very effective for consistency checking especially for complex products. Based on this, the research question could be stated as follows:

How effective is a multi-agent approach to design consistency checking, especially for distributed design of complex systems?

To answer this question we need to define a research methodology.

Research Methodology:

Research methodologies differ from discipline to discipline. In this work, we adopted an integration of quantitative and qualitative approach as a basis of this research.

The research methodology can be seen as having been conducted along four main phases. The phases are illustrated in Fig 2 and described below:

- **Phase 1:** Hypothesis definition, literature review, initial data collection
- **Phase 2:** the definitions of a general model of the mismatch control process; general principles of mismatch control; and of a general taxonomy of design mismatches. Based on the general mismatch control approach the structural multi-agent framework is developed.
- **Phase 3:** Developing of the research prototype, initial testing of research prototype,
- **Phase 4:** Application stage: The implementation of a theoretical framework for a specific industrial case study.

  During this phase a number of industrial interviews is conducted to analyse and verify specific design data.

  Data collection and analysis is conducted in a structured way and is based on unstructured and semi-structured questionnaires.

Stages of Research:

This research has applied the recent developments in the area of concurrent engineering and advanced design using concepts from

Distributed Artificial Intelligence (DAI), in particular Multi-Agent Systems (MAS). The research stages are described below.

1. Carry out an Extensive literature review about existing AI methods and tools available for Design, and current AI solutions for aerospace design support. Agent-based projects for design and Concurrent Engineering, as well as an analysis of tools for multi-agent systems development, is critically analysed with respect to their capability of handling mismatch control.

2. Devise general principles of Intelligent Distributed Mismatch Control. Develop a Methodology of Distributed Mismatch Detection – IDMC-approach.

3. Devise the general taxonomy of mismatches in Design. Devise the taxonomy of mismatches in aerospace design.

4. Develop a Conceptual Framework for a Multi-Agent System that handles mismatches. This step is a major milestone of this research. Several issues need to be considered at this level including:

-The design knowledge needed to be considered within each agent.

-The knowledge representation paradigm.

-Communication and negotiation issues, including conflict resolution.

5. Capture of the industrial requirements, knowledge acquisition and elicitation.

6. Development of a research prototype tool for mismatch control, as an initial implementation of the framework.

7. Industrial Case Study, implementation of methods and tools to a specific aerospace design process.

8. Analyse the performance of the theoretical and practical results.

The general plan of investigation is outlined in Fig. 2, including relations between stages of this research. The next section outlines the contents of this dissertation.

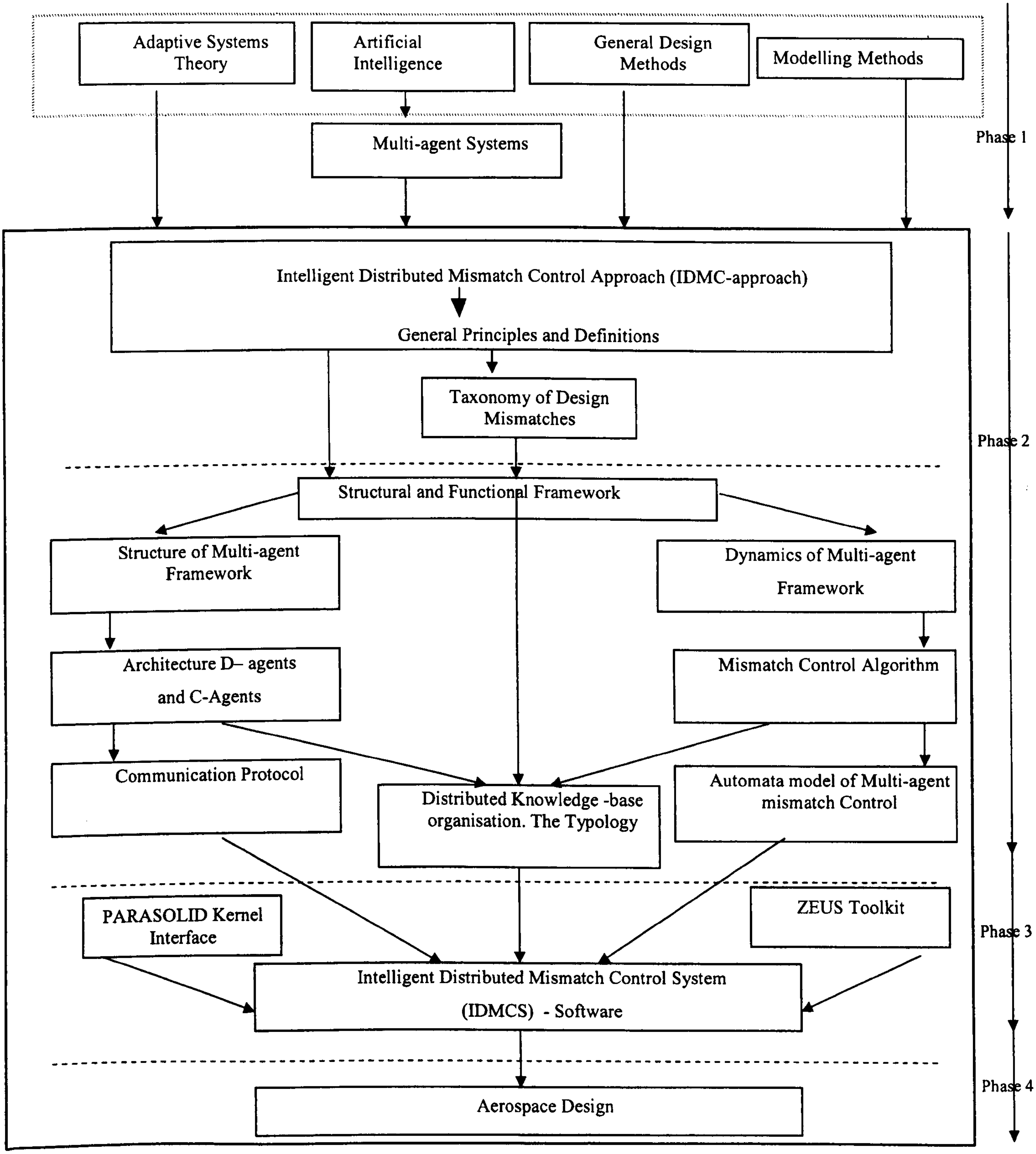

Adaptive Systems Theory
Artificial Intelligence
General Design Methods
Modelling Methods
Multi-agent Systems
Phase 1
Intelligent Distributed Mismatch Control Approach (IDMC-approach)
General Principles and Definitions
Taxonomy of Design Mismatches
Phase 2
Structural and Functional Framework
Structure of Multi-agent Framework
Dynamics of Multi-agent Framework
Architecture D– agents and C-Agents
Mismatch Control Algorithm
Communication Protocol
Distributed Knowledge -base organisation. The Typology
Automata model of Multi-agent mismatch Control
PARASOLID Kernel Interface
ZEUS Toolkit
Phase 3
Intelligent Distributed Mismatch Control System (IDMCS) - Software
Phase 4
Aerospace Design

Figure 2: General Plan of Investigation

1.5 Dissertation outline

The contents of this dissertation are given in approximately chronological order. This dissertation consists of seven main chapters (Chapters 2-8), followed by a conclusion (Chapter 9).

Chapter 2 looks at the background to the work, considering previous research into AI in Design, CE and the potential approaches, which could be used. Previous work, particularly that of commercial products and Research prototypes is then examined, shows some of the problems and benefits of metaphor in computing. The chapter also examines the different definitions of CE [Winner at al., 1988], Unan [Unan, 1992], including its role in the development frameworks. Finally, the chapter looks at some of the most important examples of Concurrent Engineering and Engineering design support.

Chapter 3 then lays out an Analysis of models of conflict Management in Design, together with proposals for the useful application of these models to Concurrent Engineering. Finally, the chapter lays out the assumptions made in the development of the model and some questions that it raises, together with proposals for testing them.

Chapter 4 describes a novel methodology of Intelligent Distributed Mismatch Control including the definition of IDMC-approach, model of design project, basic scheme of IDMC, and the general framework for development of taxonomy of design mismatches.

The application of methodology is considered in Chapter 5 and the particular Multi-agent method developed and described. In this Chapter the formal description of Multi-agent framework is presented and the structure and communication issues are introduced. Also the formal model of MAS dynamics is described. The model is based on automata formal notation and helpful to for analysis the different forms of communications between agent and designers.

The principles of organization of IDMCS are then described in Chapter 6. This Chapter looks out the main principles of IDMCS development, IDMCS architecture, and the methods of knowledge elicitation for mismatch control process.

Chapter 7 describes the principles of aerospace design and mismatch control in aeronautics. The methods of conflict resolutions in aerospace design using IDMC-approach are presented. The taxonomy of mismatches in aerospace design is presented.

Finally, Chapter 8 draws together results and conclusions from the previous tests and experiments. Suggestions are also made for further research in testing the model and applying it to other areas of Concurrent Engineering.

Chapter 2 Background to Distributed Artificial Intelligence in Engineering Design and Concurrent Engineering

2 Artificial Intelligence in Design

2.1 Current AI solution for Aerospace Design

2.1.1 Concurrent Engineering Approaches. Virtual Mock-up Software

The analysis of current applications of AI in Engineering design and Concurrent Engineering is described in this Chapter.

There are many perceptions about the nature of Concurrent Engineering (CE) [Prasad, 1995; D'Ambrosio et al., 1996; Szczerbicki, 1994; Pham and Dimov, 1998; Tong and Fitgerald, 1994; Jin, et al., 2001; Sun, Zhang, and Nee, 2000], also known as simultaneous engineering.

[Winner et. al, 1988] defines concurrent engineering as "a systematic approach to the integrated, concurrent design of products and their related processes, including manufacture and support. This approach is intended to cause the developers, from the outset, to consider all elements of the product life cycle from conception through disposal, including quality, cost, schedule, and user requirements."

According to [Unan, 1992] "Concurrent engineering is getting the right people together at the right time to identify and resolve design problems. Concurrent engineering is designing for assembly, availability, cost, customer satisfaction, maintainability, manageability, manufacturability, operability, performance, quality, risk, safety, schedule, social acceptability, and all other attributes of the product".

According to Unan and Dean [Unan and Dean, 1992], for the organisation of concurrent engineering environment, the design process should incorporate, at its various stages, a number of concurrency attributes.

Figure 3 [Szczerbicki, 1994] represents a sample of concurrency attributes.

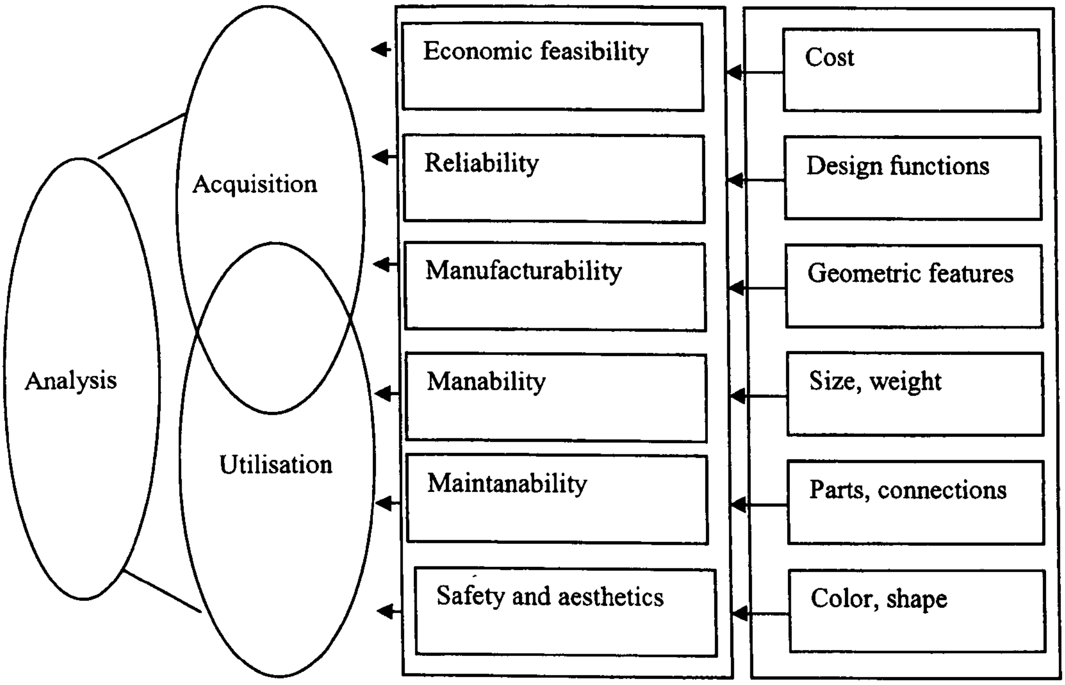


Figure 3: Sample Concurrency Attributes [Szczerbicki, 1994]

Because CE is a very complex process the analysis of relations between participants is important. Figure 4 shows the Concurrent Engineering Relations between tasks, teams, etc.

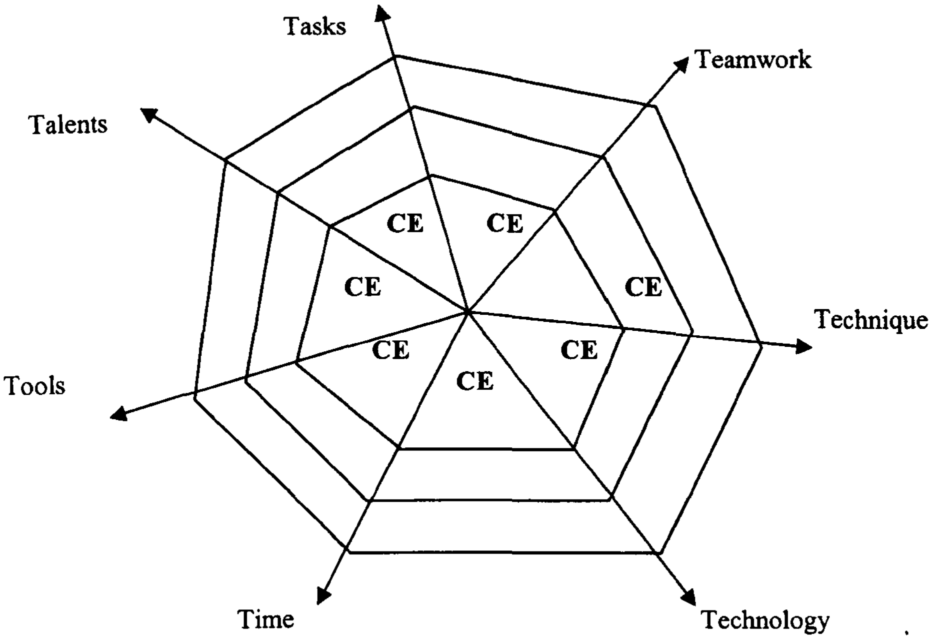


Figure 4: Concurrent Engineering Relations [Prasad, 1995]

At the present time, CE approach is strongly connected with the idea of organising not only virtual design teams [D'Ambrosio et. al, 1996], when designers are distributed between different places and/or countries, but moreover, virtual corporations - virtual factories [Davidow & Malone, 1992; O'Leary et al. 1997; Schmitt, 1996] as a future development of traditional manufacturing processes.

According to [Szczerbicki, 1994] Concurrent Engineering is a strategy that attempts to process as many product development tasks in parallel and incorporate relevant life-cycle attributes as early as possible in the design phase. The goals of CE are to reduce the duration of design projects, save development costs, and provide better quality products. The implementation of such a strategy considerably increases the complexity of the design process and makes it more difficult to plan and manage. New approaches and tools based on artificial intelligence methodologies are needed to deal with the above complexity.

Hierarchical Concurrent Engineering (HCE) [D'Ambrosio, Darr and Birmingham, 1996] is a good model of concurrent engineering that attempts to do two things: maximize concurrency in a concurrent-engineering process through decentralized, distributed decision making, and optimize through shared preference structures and constraint networks. In HCE, designers are represented as rational decision-makers that are part of a network composed of constraints and (partially) shared hierarchical preference structures. A key aspect of HCE is that it stresses decentralized decision making by designers: decentralization provides increased concurrency during the design process, makes modeling the design process easier, and has the potential to scale well. The overall activity is exploring computational methods for analyzing, synthesizing, and evaluating (the aesthetics) artistic expression. The hierarchical concurrent engineering (HCE) as a general line of research exploring alternate organizational forms and decision-making processes to support concurrent engineering is introduced.

The paper of [Jacquel et. al, 1997] describes a novel approach to the design of concurrent engineering systems by reversing the traditional view such a system as number of distinct, but integrated modules operating on a data structure that is product model.

In the publication of [Pham and Dimov, 1998] a new approach to concurrent engineering, focusing on simultaneous product design and process planning is presented. The key elements in this approach are a framework for structuring manufacturing information and maximizing the information-carrying capacity of the design models, a procedure for intelligent mapping features on to pertinent manufacturing considerations.

An approach for concurrent engineering environment is presented by Ndumu and Tah [Ndumu & Tah, 1998]. The authors have developed an agent-oriented approach, which used a FIPA protocol for agent's coordination and TCP/IP as agent's network standard.

Other European and National projects are oriented towards developing a CE approach particularly for different fields of applications, such as Aerospace [AIT Initiative] Addressing the CAD/CAM/CAE..., 1999; Hale & Craig, 1994; Hale, 1994; AIDA], Automotive, or oriented to finding general specifications and standards for distributed design, such as AIT Initiative [AIT Initiative].

In this case, the design for assembly, as a part of the CE process, is one of the most important issues [Addressing the CAD/CAM/CAE..., 1999] because the design of large–complex products for automotive or aerospace industry is not possible without using modern complex tools for full support of the virtual assembly. In this context, it is very important to analyse current software for a virtual mock-up process. Some of the main tools are presented below.

Tecoplan [Automatic design Verification, ADV V3, 1999] Automatic Design Verification (ADV) automatically uncovers all collisions and violations of minimum distances during the early design phase. ADV based on Tecoplan's formal model name Tecoplan's Space Management. The highlight is that every part knows its environment, and only relevant parts are checked against each other, automatically. All errors will be corrected before building the first real prototype. The functionality of ADV includes:

- Automatic Collisions and Minimum Distance Analysis,
- Assembly Checks,
- Dynamic Multi-part Simulation,
- Engineering on the Web.

dV/MockUp [Buyer's Guide, Mechanical CAD..., 1999] is a family of tools for Interactive Product Simulation - the process by which design and manufacturing companies can study the form, fit and function of their products.

The largest assemblies (100,000s of parts) can be imported into dV/MockUp allowing the designer to work in the context of the overall design. dV/MockUp's technology provides real-time visualisation and interaction with even the largest of assemblies, allowing picking and moving parts interactively. The user has access to the entire product structure, controlling sub-assemblies, selected parts and assemblies. Central to dV/MockUp, is a multi-process, real-time discrete-event simulation engine.

Other systems such as CATIA, IDEAS are described in [Buyer's Guide, Mechanical CAD..., 1999]. A brief comparison between Virtual Mock-up Software characteristics is depicted in Table 1.

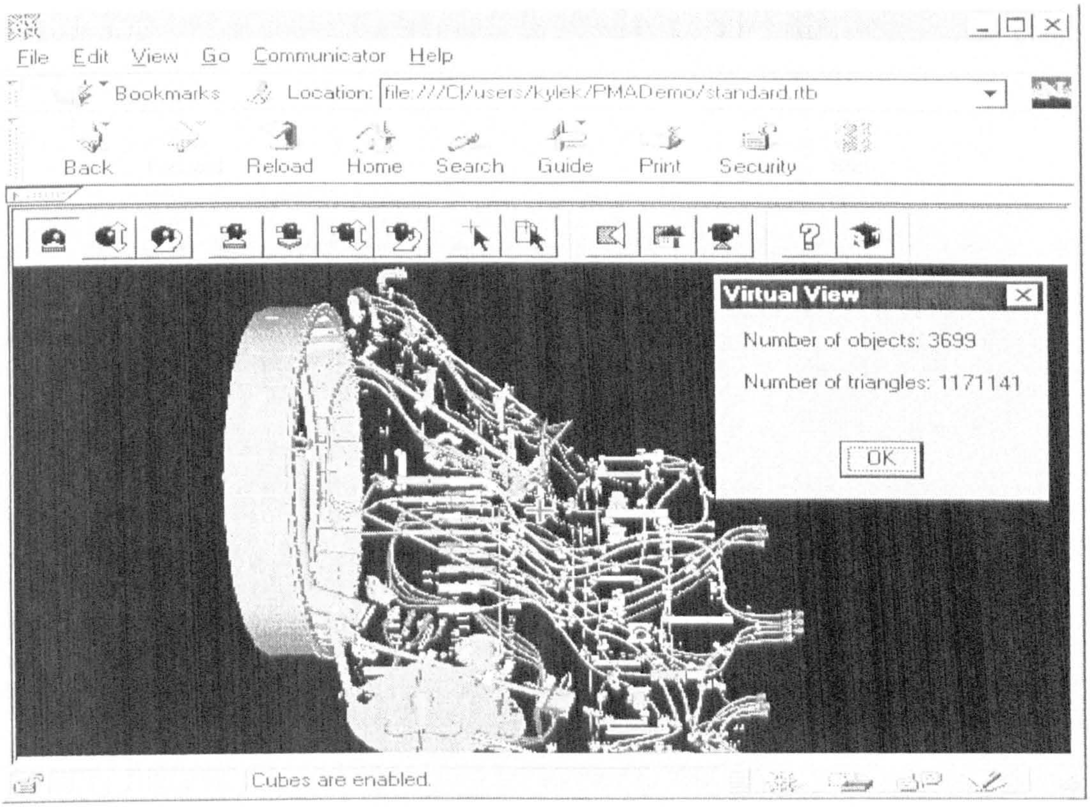
Table 1: Virtual Mock-up Systems

Name of Project	**Type of System**	**Types of mismatches**	**Web Integration**	**Additions**
DMU CATIA	Commercial	Inference	Yes	Possibility of integration with expert systems
A D V	Commercial	Sequences,	Yes	-

Name of Project	**Type of System**	**Types of mismatches**	**Web Integration**	**Additions**
		collisions, minimum distance		
dV/Mock-up	Commercial	Interference	Yes	-
IDEAS	Commercial	Interference	Yes	-
SCOPES	Academic prototype	Mismatches of configuration	No data	Application: tolerant manipulators
VirtualView	Commercial	Interference, Tolerance	Yes	Assembly

The commercial Virtual Mock-up systems described above are very powerful tools. However, the current virtual mock-up software, in general, do not use any AI techniques for detecting and resolving design mismatches. The systems can detect geometric inconsistencies based on syntax level, but have no capability of advising how to change the design project in order to meet design requirements.

As can be seen, the result of Interference Detection, only for eight parts of assembly, is difficult to analyse, but normally assemblies for aerospace or automotive industry include about 1000-10000 elements (see Fig. 5) Virtual View [Virtual View, 1996].



Location: file:///C|/users/kylek/PMADemo/standard.rtb

Number of objects: 3699

Number of triangles: 1171141

Cubes are enabled.

Figure 5: VirtualView. An example of industrial assembly (© Resolution Technologies Inc.)

The following section of this report describes some of the important developments that have been made to date in the field using AI methods for Aerospace Design.

2.1.2 AI based Projects for Aerospace Design. Enterprise Integration

The research Project named AEROEXTN [Sheldev et al., 1996] started in June 1997 with Cranfield University and the University of Luton as co-investigators. The major industrial collaborator had a successful Concurrent Engineering process that was working well with in-house manufacturing. With the advent of outsourcing, they wanted to extend their CE process to the supplier base. According to the authors, the purpose of this project is to develop processes by which competence in Concurrent Engineering can be developed into the Extended Enterprise with advantage to quality, time, and cost competitiveness. Because the capacity, capability, skills and resources of suppliers will differ, decisions have to be made as to the extent to which any supplier can be incorporated in the CE loop and the impact on the processes of the different decisions.

The complexity of the problem arises in the issues of:

- integration of IPR/contractual expense;
- risk and reward;
- relationships of trust and cultural fit;
- IT matters of data management, hardware and software in the demanding aerospace manufacture environment of rigorous configuration control and change management.

Ilan Kroo et al. [Kroo and Takai, 1988] from Stanford University present a Quasi-Procedural, Knowledge-Based System for Aircraft Design. This work deals with the development of a program for aircraft design, combining a rule-based advice and warning system with an extensible set of routines in an unconventional architecture. The system consists of several procedural modules for calculation of aircraft aerodynamics, structures, propulsion, and operating costs, which, when executed in the appropriate order; permit computation of desired results. This structure is encapsulated in an executive routine with a highly interactive, event-driven, graphical interface and expert system. The rule-based system is used to assist the user in selecting intelligent design solutions and appropriate procedures.

Shedev et al. [Shedev et al., 1995] have described a Design for manufacture *method* applied for aerospace industry. This paper is a result of the first phase of a study conducted by Cranfield University to establish the user requirements for "design for manufacture" with in a complex design and manufacture supply chain.

Wallace [Wallace, 1996; Wallace & Sackett, 1996] describes a SCOPES project - Systematic Concurrent design of products, Equipment's and control Systems). This paper presents the results of a three-year project to develop a suite of integrated software modules, which enable design support on the downstream functions associated with the assembly of mechanical and electromechanical products throughout the design process. The project is geared for Digital Mock-Up of large products (Boeing 777) and uses CATIA design system.

Williams et al. [Williams et al., 1999] describe a composite design software which reduces engineering time of Eurofighter parts. FiberSIM simulation software allows manufacturing engineers to define composite characteristics and reduce design cycle time.

Leiening and Blount [Leining & Blount, 1998] have described the implementation of knowledge-based engineering (KBE) for aircraft wheel and brake industry. The paper investigates tools to increase productivity, explains the way a KBE tool works, and describes possible KBE applications as design and diagnostic tool.

Benett [Bennet, 1997] describes an application of virtual prototyping in development of complex aerospace products. The work considers the development and application of a comprehensive virtual prototyping initiative applied to the mechanical design and manufacture domain at British aerospace.

Pan, et al [Pan et. al, 1997] have described an advanced CSCW technology which has a great impact on the communication and cooperation during the design process of products. In this paper authors pay particular attention to the cooperative behavior and management in distributed collaborative design systems. A prototype is presented.

Hale and Graid [Hale and Graid, 1994, Hale, 1994] have developed a distributed intelligent system for aircraft design based on conception of a design integration framework. An Intelligent Multi-disciplinary Aircraft Generation Environment (IMAGE) is described which uses state-of-the-art-computing technologies.

Subbu et al [Subbu et. al, 1998] present a Virtual Design Environment to support design-manufacturing-supplier planning decisions in a distributed, heterogeneous environment. The approach utilizes evolutionary intelligent agents as program entities, which generate and execute queries among distributed computing applications and databases. A prototype of Virtual Design Environment has been implemented using CORBA [CORBA, Common Object Broker Architecture] as principal distributed systems programming tool.

Mullins and Anderson [Mullins & Anderson, 1998] describe a graph-based approach for automatic identification of geometric constants in mechanical assemblies. They present a new technique for the automatic identification of such constraints in 3D assemblies with no orthogonal contacts between component surfaces and kinematics joints. Search algorithms for identifying assembly constraints in these graphs are presented.

The Design Process in Aerospace Industry Project [Design Process in the Aerospace Industry...] is a three year program funded by the EPSRC Innovative Manufacturing Initiative. The main themes of the research are: to identify sources of error within design process, and propose and evaluate design process changes to reduce error incidence; to create and evaluate measures of design quality and their interactive use in increasing the rate of convergence to design objectives; to determine how best to integrate all data flows in the design process, with particular emphasis on inputs to and from specialist knowledge and methods skill group. Table 2 shows a summary of the main characteristics of the systems mentioned above.

Table 2: Current AI and CE based approaches for Aerospace Design

Name of Project	**Authors**	**Type of System**	**Knowledge Representation Paradigm**	**The field of implementation**	**The level of realisation**	**Additions**
Kvasi procedural, Knowledge-based System	Kroo et al.	Advice	Rules	Aircraft Design	Prototype	-
SCOPES	Wallace	Advice	Rules	Boeing 777 Design, Design for Assembly	Prototype	Uses together with CATIA
FiberSIM	Williams	Composite design	No knowledge-base	Composite Design for Eurofighter	Commercial	Uses together with CATIA

Name of Project	**Authors**	**Type of System**	**Knowledge Representation Paradigm**	**The field of implementation**	**The level of realisation**	**Additions**
				Parts		and EDS Unigraphics
Knowledge-base Engineering	Liening	Diagnostics	Expert system, No data	Aircraft wheel and brakes industry	Direct realisation	-
IMAGE	Hale	Advising	Agents based Rules and frames	Multi-disciplinary Design	Prototype	CATIA
Virtual design Environment	Subbu	Environment	Evolutionary agents	Virtual design	No data	-

The analysis of implementation of AI methods for aerospace design suggests that current aerospace design support systems are highly specialised and use different approaches such as expert systems, multi-agent systems and intelligent interfaces. The systems are implemented as advice systems.

Unfortunately, there are very few software tools available on commercial levels. Some systems do provide appropriate facilities for distributed design however, but the differences between internal models of knowledge analysis and representations of their systems are obviously restricted their implementation for Industry applications.

What is also clear is that, at present, there are no fully integrated intelligent design aerospace systems available for checking mismatches and automatic modification of the design project.

2.2 Agent-based projects for Design and Concurrent Engineering. Enterprise Integration

2.2.1 Agent Definition

Defining what is an 'agent' and a 'multi-agent system' is important. There are many definitions of agents and multi-agent systems that can be found in [Brooks, 1990; Hale & Craig, 1995, 1996; Kaelbling & Rosenschein, 1990; Nwana, 1996; Nwana and Ndumu, 1999; Hyacinth and Nwana, 1998; Shen and Norrie, 1999]. For the purpose of this work one of the most accurate definitions is presented by Sycara [Sycara, 1998]. There the agent is a system with characteristics as follows:

Situatedness means that the agent receives some form of sensory input from its environment, and it performs some action that changes its environment in some way.

Autonomy means that the agent can act without direct intervention by humans or other agents that has control over its actions and internal state.

Adaptivity means that an agent is capable of (1) reacting flexibly to changes in its environment; (2) taking goal-directed initiative, when appropriate; and (3) learning from its own experience, its environment, and interaction with others.

Sociability means that an agent is capable of interacting in a peer-to-peer manner with other agents or humans.

According to Sycara that these four properties uniquely characterise an agent as opposed to related software paradigms, such as object-oriented systems, or expert systems. A more detailed discussion about comparison between agents and object-oriented systems can be found in [Jennings et. al., 1998].

In this research we will use Sycara's definition as most accurate and focused on social capability of agents.

2.2.2 Architecture of agent

There are a numerous architectures described for multi-agent frameworks [Brooks, 1990; Brown & Grecu, 1997; Ferber, 1999; Kaelbling & Rosenschein, 1990; Sycara, 1998; Kaelbling et. al., 1995; Yongtong et. al., 1996]. For different types of agents (cognitive, reactive, hybrid) and different applications (systems simulation, artificial life, intelligent control, problems solving etc.) are implemented different types of architectures.

Typically [Sycara, 1998], multi-agent architectures for problem solving applications are realised as a number of software layers, each dealing with a different level of abstraction.

Most of architectures are represented as three layers. At the lowest level in the hierarchy, there is typically a reactive layer. The middle layer typically abstract away from raw sensor input and deals with a knowledge-level view of the agent's environment, typically making use of symbolic representation. The uppermost level of the architecture tends to deal with the social aspects of the environment.

Coordination with other agents is typically represented in the uppermost layer.

In this research, we will attempt to identify the most suited architecture for mismatch control in design. In this case our framework will be a software apparatus for problem solving in distributed environment.

2.2.3 Languages of multi-agent systems

Multi-agent systems are computing programs and cannot be designed without using a set of languages for description, realisation of structures and procedures. According to [Duffy & Andreasen, 1999] we can define five languages which will be necessary to design and realisation of a multi-agent system. Languages are described below.

Implementation languages – L1

These languages are used for programming of multi-agent systems. These languages are usually the classic programming languages as Lisp, C/C++, Java, or Smalltalk or special agent-programming extensions of these languages, such as Telescript [Telescript Language], AKL [Agents Kernel Language], Python [The Pithon Programming Language], AGENT_CLIPS [Cengeloglu, 1995].

Communication Languages – L2

These languages provide interaction between agents by means of data transmissions and reciprocal requests for information and services.

At the present time a general standards for these languages are established. Examples of these standard languages are KQML, FIPA, and etc.

Languages for describing behaviour and the laws of the environment - L3

Using these languages it is possible to define what is happening in multi-agent framework in abstract manner.

It will be possible to analyse algorithms of communications, negotiations and conflict resolutions. For formalisation of abstractions represented these languages are using, for instance, productions, automata theory, Petri-nets [Cost et. al., 1999; Xu and Deng, 2000; Fernandes and Belo, 1998; Holvoet, 1995], DEVS formalism, Markov chains, and algebraic/language-based models [Inat and Varaiya, 1989; Gohen et. al.,1989].

Languages for representing knowledge – L4

These languages are used for describing internal models of worlds for cognitive agents or these combinations with reactive agents. It may be rule-based languages, frames, semantic nets, predicates, and combination of these approaches.

Formalisation and specification language - L5

These languages are used for describing multi-agent systems at the most abstract level, for instance, connections with other external languages or programs, to describe some additional meta-parameters of MAS, notes about interactions, etc.

Later in this report, we will come back to these languages when describing how they are used for design and realisation of intelligent distributed mismatch control approach. The same numeration of Languages L1, L2, ..., L5 is used for the representation of multi-agent system's stages of development. In our concern, the languages L1,..., L4 are necessary for definition of MAS, and L5 can be used as an additional description language.

The next sub-part is represented a key publications in the field of multi-agents systems in design.

2.2.4 Multi-agent systems in design and Concurrent Engineering. Key publications

Many of the recent developments in the field of Agents and AI for Design have been investigated and described by Brown and Grecu [Brown and Grecu, 1996, 1997], Bento and Fejo [Bento & Feijo, 1997; Bento, 1998, 2000], D'Ambrosio [D'Ambrosio et. al, 1996], and Ndumu and Nwana [Nwana, 1996; Ndumu & Tah, 1998].

The research report [Brown & Grecu, 1997] provides a useful introduction to the study of AI in Design and goes on to describe a number of potential fields of applications. In their more recent work Bento and Fejo [Bento & Feijo, 1997], present an agent-based paradigm for building Intelligent Computer Aided Geometric Design systems using a predicates-based distributed knowledge-base. The geometric kernel is ACIS.

D'Ambrosio [D'Ambrosio et. al., 1996] presents a hierarchical concurrent engineering model based on agent methodology. This research is more concerned with developing the theoretical framework that can be used to create CE agent-based systems.

Ndumu and Nwana [Ndumu and Nwana, 1999] described the implementation of ZEUS toolkit for organisation of agent-based Computer-Assisted Collaborative Design.

These and other approaches for implementation of multi-agent systems in Design are described in Table 3.

Table 3: Agent based Design Projects

Name of Project	**Authors**	**Type of System**	**Agents types**	**The stages of design implementation**	**The level of realisation**	**Web integration**
HCE	D'Ambrosio	Framework for support concurrent engineering	Adaptive	Design, CE	model	no
No data	Vittikh, Skobelev [Vittikh, Skobelev, 1998]	Framework of self-organised and cooperation processes	Self-learning	Complex Systems Analysis	model	no
No data	Schmitt	Support of Virtual Environment	No data	Designer-Machine Interaction	model	yes
XLOG+	Fejo Bento	Computational support	Hybrid agent architecture, object oriented, predicate logic	Assembly, Mock-up	model	yes
PACT	Cutkosky	Concurrent Engineering environment	Predicates, first order logic	Simulation, Distributed redesign, CAE	prototype	yes

Name of Project	**Authors**	**Type of System**	**Agents types**	**The stages of design implementation**	**The level of realisation**	**Web integra-tion**
MAGSY	Fischer [Fischer, 1994]	Design of Manufacturing Systems, shop floor control	Hierarchical planning structure	Planning and control in flexible manufacturing systems	prototype	no
SiFA	Brown Grecu	Model of learning in Agent-based design	Learning agents, single function agents	Agent-based design methodology	Model and prototype	no
DESIRE	Brazier [Brazier et.al. 1990, 1995]	Collaborative engineering support	No data	No data	No data	No data
REDUX	Petrie	Internet oriented agents architecture "intelligent web agents"	KQML support agents Web oriented	Agent-based Engineering	Model	Yes, Web oriented agent architect ure
No data	Paderis	Advice and design	No data	Design of rapidly deployable fault tolerant manipulators	Research prototype	Yes
Cdb	Varma	Design support	Designer – supported agents	Web based design	No data	No data
TEAM,	Prasad	Learning model	Heterogeneo	Cooperative	Model and	yes

Name of Project	**Authors**	**Type of System**	**Agents types**	**The stages of design implementation**	**The level of realisation**	**Web integration**
L-TEAM	Lander, [Lander, 1994]	of multi-agent system Steam condensed design	us reusable, learning enabled agents	distributed search	prototype	

Unfortunately, none of the aforementioned applications of multi-agent systems is orientated towards dealing with the problem of detecting mismatches that may occur during the integration phase of distributed design.

The next sections describe a possible definition of an implementation language, for the development of Multi-agent systems and solid modelling support in CAD, particularly for IDMC development.

2.3 Software approaches for multi-agent systems development

2.3.1 ZEUS Toolkit (Language L1)

The current situation for developing multi-agent systems is such that no particular programming tools, support fully agent oriented programming. New systems were developed that use a range of agents oriented methods but these cannot be said to be agent-oriented languages.

Another way is to extend a traditional programming language to support agent - programming. In this case Java Language– can be viewed as the best platform for organisation of multi-agent software, because Java do provide a complex Internet programming support, interplatforms specification and in general very similar to C++.

The ZEUS toolkit was developed by BT Research Lab of multi-agent systems [Nwana & Ndumu, 1999; Hyacinth & Nwana, 1998; Nwana, 1996; Ndumu & Tah, 1998]. A description of the implementation of ZEUS toolkit for organisation of collaborative environment for design is given in [Nwana & Ndumu, 1999].

The ZEUS approach consists of Design and Realisation activities plus runtime support facilities that enable the developer to debug and analyse their implementations.

The toolkit [Nwana & Ndumu, 1999] comprises a suite of Java classes which help users to develop agent-based applications by integrating and extending some predefined classes.

The toolkit provides classes that implement generic agent functionality such as communication, co-ordination, planning, scheduling, task execution and monitoring and exception handling. Developers are to provide the code that implements the agents' domain-level problem solving abilities. The main components of the toolkit include:

- an agent component library,
- a set of visualization tools, and
- an agent building environment which also includes an
- automatic agent code generator.

The toolkit also provides utility agents such as a name server and a facilitator for use in knowledge discovery. The architecture of Zeus agent is represented as follows (Fig. 6) [ZEUS, 1999]:

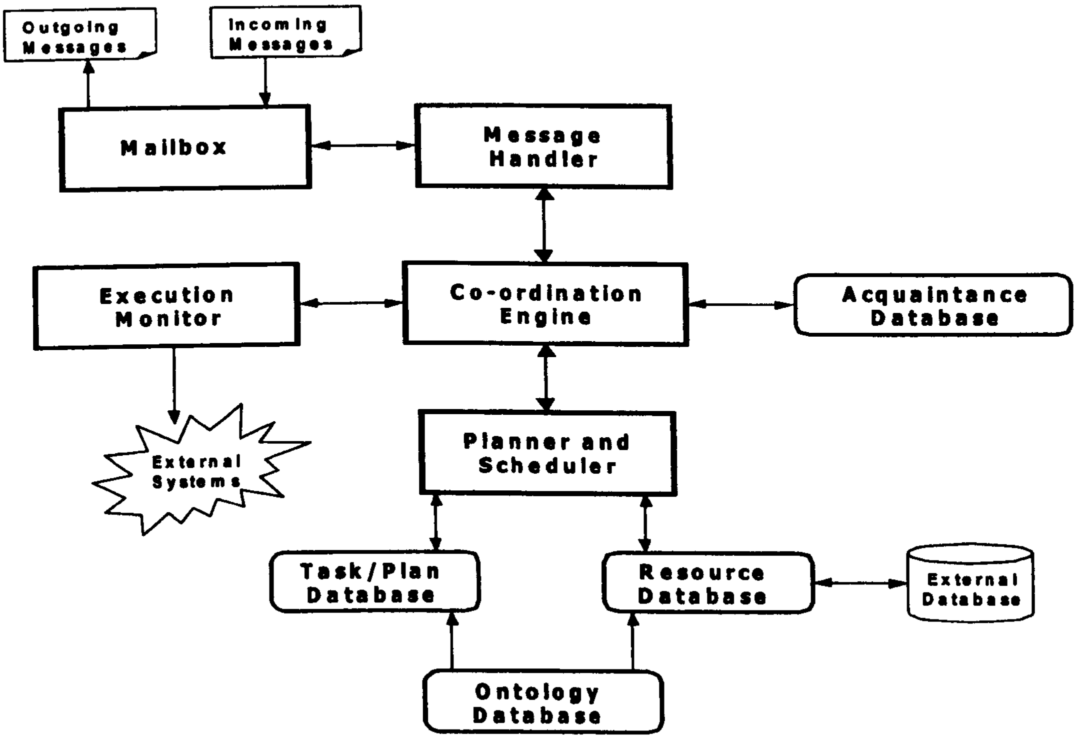


Figure 6: Architecture of the generic ZEUS agent

- Mailbox that handles communications between the agent and other agents.

- Message Handler that processes incoming messages from the Mailbox, dispatching them to the relevant components of the agent.

- Co-ordination Engine that makes decisions concerning the agent's goals, e.g. how they should be pursued, when to abandon them, etc. It is also responsible for co-ordinating the agent's interactions with other agents using its known co-ordination protocols and strategies, e.g. the various auction protocols or the contract net protocol.

- Acquaintance Database that describes the agent's relationships with other agents in the society, and its beliefs about the capabilities of those agents. The Co-ordination Engine uses information contained in this database when making collaborative arrangements with other agents.

- Planner and Scheduler that plans the agent's tasks based on decisions taken by the Co-ordination Engine and the resources and task specifications available to the agent.

- Resource Database that maintains a list of resources (referred as *facts*) that are owned by and available to the agent. The Resource Database also supports a direct interface to external systems, which allows it to dynamically link to and utilise proprietary databases.

- Ontology Database that stores the logical definition of each fact type — its legal attributes, the range of legal values for each attribute, any constraints between attribute values, and any relationships between the attributes of the fact and other facts.

- Task/Plan Database that provides logical descriptions of planning operators (or tasks) known to the agent.

- Execution Monitor that maintains the agent's internal clock, and starts, stops and monitors tasks that have been scheduled for execution or termination by the Planner/Scheduler. It also informs the Planner of successful and exceptional terminating conditions of the tasks it is monitoring. In order to manage tasks, the Execution Monitor also has a direct interface to external systems. It is assumed that the domain realisations of tasks are external programs.

The ZEUS toolkit also provides, among others, an Ontology Editor for defining the shared domain ontology and a Task Editor for describing the planning operators and reaction scripts for the agent.

Normally, agents communicate using agent's communication languages (ACLs). Most agent communication languages (ACLs) are based on speech act theory [Woldbridge, 2000], wherein human utterances are viewed as actions in the sense of actions performed in the everyday physical world (e.g. picking up a block). Hence, ACLs specify message types called performatives, such as ask, tell, or achieve, which by virtue of being sent from one agent to another, are assumed to effect some illocutionary actions in the receiving agent.

Obviously, inter-agent compatibility will be impossible until all parties adopt the same agent communication language, and fortunately ACL standards do exist. ZEUS agents communicate using messages that obey the FIPA 1997 ACL specification, which is described in htpp://www.fipa.org. This syntax is used to construct instances of the performative class, which have the following attributes [ZEUS, 1999]:

```
Performative ( type: /* performative type, e.g. inform, cancel etc. */
               sender: /* name of agent sending message */
               receiver:/* name of intended recipient agent */
               reply_with: /* sender's conversation identification key */
```

in_reply_to: /* recipient's conversation key */
content: /* message content */
language: /* name of language in which content is expressed */
address: /* sender's address */
send_time: /* time at which message is sent */
receive_time: /* time when message is received */ :)

To compare ZEUS and other agent programming languages, we can see that these are clearly restricted in their integration capability.

For instance, TeleScript system especially oriented to support only open market strategies, Python - parallel programming, Agents Clips - Clips based language.

In this case the decision in using ZEUS toolkit as one of the most fulfilled systems for agent oriented programming is justified.

2.4 Representation of Geometric Design Information.

PARASOLID Kernel

PARASOLID is a geometric kernel, developed by EDS Unigraphics [Paraphase PARASOLID, 1995; PARASOLID General Information].

To assist in the integration of PARASOLID into an application the following tools are provided: resource library - sample, annotated code for all the integration tasks; extremely thorough, on-line documentation in HTML format; attributes and groups functionality to attach data to solid models and their entities; bulletin board to track details of model changes; session rollback for permit unlimited undo and redo operations in a session; feature history management using partitioned rollback; session journaling and replay; Frustum for file handling and memory management; part storage in text and machine-independent binary format; KID (Kernel Interface Driver) with graphics for easy application prototyping.

PARASOLID is used by a wide range of companies that need to create and manipulate mathematical models of real objects. The typical applications are Computer Aided Design - CAD systems to create mathematical models based on user input. Individual models can then be combined as components of an assembly to create a whole product such as an aircraft or fax machine. PARASOLID adds value by being able to model real life objects very accurately and reliably.

Other applications include Computer Aided Engineering, Computer Aided Manufacturing, Translators, Architecture, Aerospace Design and Construction.

PARASOLID is widely accepted as *de facto* standard for design of complex systems. Such systems as SolidWorks [Buyer's Guide, 1999; Richardson, 1999], EDS Unigraphics [Addressing the CAD/CAM/CAE..., 1999; Paraphase PARASOLID, 1995; PARASOLID General Information], Solid Edge[Solid Edge, 1999] are using PARASOLID as a main geometric kernel.

2.5 Conclusion

From the literature reviewed, it is clear that AI techniques are widely used in design and Concurrent Engineering tools.

However, current systems and models are not fully supporting detection of complex design inconsistencies, particularly at the distributed design process. The use of intelligent agents as independent distributed knowledge entries promises to provide the missing link.

The investigation of methods and principles of organisation of multi-agent system for mismatch design will be investigated. This multi-agent architecture will be at the heart of an intelligent distributed mismatch control system (IDMCS) that aims at ensuring that the overall design is consistent and acceptable to all.

The problem becomes that of creating a conceptual approach for building intelligent mismatch control systems – Intelligent Distributed Mismatch Control approach (IDMC-approach). IDMC-approach should define:

- A conceptual model of distributed mismatch detection-IDMC-approach.
- A definition of mismatches, requirements for organization of models.
- A general mismatch control scheme.
- A definition a taxonomy of distributed design mismatches.
- A multi-agent framework for distributed mismatch control using an IDMC approach method.
- The framework will be based on a community of agents, which are capable of learning and/or adapting actions.

- develop a Conceptual Framework for a multi-agent system that handles these mismatches. This should take into account:
  - the design knowledge needed to be considered within each agent.
  - the knowledge representation paradigm.
  - communication and negotiation issues, including conflict resolution, adaptation (and possible, learning) strategies.
  - IDMCS- architecture.
  - implementation IDMCS for aerospace design.
  - developing and evaluation of prototype

The main proposed results of implementation of IDMC will be reducing design cost; reducing time of design; and raising professional levels of designers.

These characteristics will be analysed on the stage of evaluation of prototype of IDMCS.

Chapter 3 Background to Conflict Management in Design

3.1 Conflicts, Mismatches and Inconsistencies

Computer Supported Collaborative Work (CSCW) as a part of distributed collaborative design process and Concurrent Engineering [Prasad, 1995; Unan and Dean, 1992, Bechkoum, 1997] promise to resolve most of the difficulties above by replacing the paper and tape and physical meetings based methods by electronic communication and electronic meetings and provide a basis for virtual design environment [Prasad, 1995; Matta and Cointe at al, 1997; Sycara, 1998; Regli, 1997; Sriram, 1993].

This is more important, because for many years the design and manufacture of major European complex products, such as sattelites, airplanes and cars has been distributed across the continent. As a result of Globalization and future distribution of design and manufacturing facilities, the cooperation amongst partners is more challenging in that the design process tends to be sequential and requires centralised planning teams and/or a great deal of travel to/from distributed designers.

In a virtual team designers work together and use the Internet/Intranet for communication. The design is a multi-disciplinary task that involves several stages. These stages include input data analysis, conceptual design, basic structural design, detail design, production design, manufacturing processes analysis, and documentation.

In general, the virtual team is very changeable in terms of designer's participation and, moreover, the environment itself changes over time. As a result the number of design mismatches can increase significantly.

The methodology of Computer Supported Collaborative Design (CSCD) is needed for future progress. In this thesis we focus on one aspect of CSCD: mismatch control during the detail design stage.

The mismatch detection during detail design stage is one of the most important, because mismatches in the early stages of detail design will have a direct implication on cost of the product, particularly for large–complex products, particular in aeronautics or automotive sectors.

The next section of this thesis briefly reviews a current models and methods in Computer Supported Collaborative Work (CSCW) and consistency checking.

3.2 Methods and models of consistency checking

Many of the recent developments in the field of conflict management have been investigated and described by Matta, Lander, Klein and others [Matta, 1996; Matta and Cointe, 1997; Lander, 1994, Klein, 1991, 1992, 1995; Easterbrook, 1991; Bechkoum and Taratoukhine, 1999a, 1999b; Grasso, 1998; Gupta et al, 1996; Mukhopadhyay and Gupta, 1998; Nuseibeh, et al., 2000; Volker, 1999].

We can say that CSCW approaches look at conflicts that occur through coordination breakdowns and are resolved through group harmonization techniques that involve articulating conflicts. Group harmonization techniques include problem-structuring methods; design rationale; immersive practices such as participatory design; conversational props; abstraction and summarization; report writing, and etc.

According to [Lander, 1994] there are several ways in which conflict can be managed: such as Avoidance - Avoid conflict by sharing information about local constraints and priorities; Conflict classification - Build taxonomy of conflict types. Associated with each conflict type is a specific piece of conflict resolution advice; Negotiation - Techniques in this area include bargaining, restructuring, constraint relaxation, mediation, and arbitration.

Below a developments of conflict management methods are represented.

[Klein, 1992] represents a conflict management as an exception-handling component of a collaborative design tool.

[Matta, 1996] defines a library of associations between concurrent engineering sub-tasks and conflict management methods to guide an agent to determine appropriate methods to manage conflict in a particular application

[Easterbrook et al, 1993] explores the support for conflict management in CSCW tools. And identifies broad areas in which conflict and conflict resolution have been studied.

[Castelfranchi, 1996] presents conflict ontology as set in a competitive situation. This conflict ontology is based in the social sciences, which needs to be expanded before it can be of any real use.

The analysis of current development in conflict management for CSCW suggests that most of these methods and frameworks are paid more attention to social and psychological aspects of communications between members of team, but not to problems of communications between artificial agents and development a general methodology of conflict management/intelligent control, based on Distributed AI.

3.3 Conflicts in Multi-agent Design systems

There are a number projects described the conflict detection and resolution methods in agent and multi-agent frameworks. Generally, MAS conflict management approaches concern conflicts between software agents, but not conflict management between designers.

In this case interesting work conducted by Yan Jin [Jin et al., 2001]. In this research the decision based approach to model of design process is presented. The notion of design values was presented. The agent-based decision network (ADN) to support concurrent decision making and collaboration in design is developed. According to authors the results are indicated that ADN increases the efficiency and effectiveness of the design process. The MAS systems co-ordination taxonomy is presented. Unfortunately the approach is not reviewed any particular case studies so it is difficult to say about effectiveness of system based only on this paper.

The research of Kuchar and Yang [Kuchar and Yang, 1997] from Department of Aeronautics and Astronautics Massachusetts Institute of Technology is about the design and evaluation of traffic conflict detection and resolution systems which requires the use of analytical models that describe encounter dynamics and the costs and benefits of avoidance actions. A number of such models have been applied in the past to the problem, but there has been no cohesive discussion or comparative evaluation of these approaches.

According to [Wagner et al, 1999] conflict in multi-agent systems is ubiquitous. Research often focuses on the process of resolving conflicts between different agents - the inter-agent conflict resolution process. However, in complex problem solving agents the process of resolving conflicts with other agents impacts local problem solving as well as deals made with other agents. This leads to the need for an intra-agent conflict resolution process between the agent's coordination mechanism and its local controller.

Examples of work in conflict management include [Klein, 1996] and [Matta, 1996]. Klein views conflict management as an exception-handling component of a collaborative design tool. Matta develops a library of associations between concurrent engineering sub-tasks and conflict management methods to guide an agent to determine appropriate methods to manage conflict in a particular application. Both Klein and Matta manage conflict by classifying.

MAS technology involves coordinating the activities of intelligent, semi-autonomous software agents. Conflict management becomes important when the environment changes over time and agents have to adapt their coordination strategies. Conflicts are managed by sharing information about local constraints and priorities; incorporating a taxonomy of conflict types and conflict resolution actions into MAS; or by negotiation algorithms.

Table 4 shows a summary of the methods of conflict management.

Table 4: Methods of conflict management

Authors	**Description**	**Implementation. Case Study**
[Lander, 1999]	General classification of conflict management methods: Avoidance; Conflict classification; Negotiation.	No information
[Klein, 1992]	Conflict management as an exception-handling component of a collaborative design tool.	No information
[Matta, 1996]	Library of associations between concurrent engineering sub-tasks and conflict management methods to guide an agent to determine appropriate methods to manage conflict in a	CSCW

Authors	**Description**	**Implementation. Case Study**
	particular application	
[Easterbrook, 1993]	Support for conflict management in CSCW tools.	Software engineering, CSCW
[Castelfranchi, 1996]	Conflict ontology as set in a competitive situation	Social Sciences
[Jin at.al., 2001]	The notion of design values was presented. The agent-based decision network to support concurrent decision making and collaboration in design is developed.	No information
[Kuchar and Yang, 1997]	Traffic conflict detection and resolution systems. Analytical models that describe encounter dynamics and the costs and benefits of avoidance actions	Traffic management
[Appelbaum., S., et al. 1999]	A conflict resolution analysis in self-directed teams	Management Sciences
[CERL project]	CERL and Impact Lab are directed by Dr. Jin, University of South California. Socio-technical Framework for Conflict Management in Collaborative Design. Conflict Management Strategy	Collaborative Design Software prototype development
[Cointe, C., Matta, N., Ribire, M. 1997]	Design Propositions Evaluation: Using Viewpoint to manage Conflicts in CREoPS	CREoPS research project
[Vagner et al., 1999]	Process of resolving conflicts between different agents - the inter-agent conflict resolution process	Multi-agent systems

3.4 Conclusion

In this part of dissertation the methods and approaches for conflict management in CSCW and design have been described and analysed.

Current models and methods in Computer Supported Collaborative Work (CSCW) and consistency checking are briefly reviewed.

In first part the description of conflict, mismatches and inconsistencies has been presented.

In second part, a number research projects have described. The conflict detection and resolution methods and taxonomies of conflicts have been reviewed.

Finally, in third part the methods and models of consistency checking are analysed. The summary of the methods of conflict management have been presented (Table 4), including papers of Lander, Klein, Matta, Easterbrook, Castelfranchi, and others.

The methodology of Distributed Mismatch Detection is needed for future progress.

The next session will describe the Methodology of Distributed Mismatch Detection in Design - Intelligent Mismatch Control approach.

Chapter 4 The Methodology of Distributed Mismatch Detection in Design

This section of the report describes and discusses the Methodology of Distributed Mismatch Detection in Design. The material that is presented is divided into three sub-sections. The first of these describes the Intelligent Distributed Mismatch Control and outlines IDMC-approach. Taxonomy of design mismatches is then presented; formal description of structure of multi-agent framework, cooperation and negotiation, dynamics of multi-agent framework and distributed knowledge-base organisation are described.

The second part discusses the organisation of the Intelligent Distributed Mismatch Control System and system's development and overview. The third part describes an implementation of IDMCS for aerospace design-the wing assembly process is described, as well as, IDMCS agent's tasks, social and domain responsibilities.

4.1 Intelligent Distributed Mismatch Control approach (IDMC-approach). Definition and Methodology

In this section I will give some important definitions and describe general principles of Intelligent Distributed Mismatch Control. It is clear, that design is a multi-disciplinary process that involves several stages such as conceptual design, basic structural design, detail design, production design, manufacturing processes analysis, and documentation. Different types of mismatches of detail design stage restrict intelligent Control of mismatches in this research.

To define a general principle of IDMC, firstly we will describe what is design project, design mismatches, describe types of mismatches and model knowledge about design project.

4.1.1 Model of design project

In general, the model of design project is included a set of elements of assembly. Each element is represented as set of structure characteristics and set of parameters. So we have M_1 – model of system.

$M_1 = < M_{str}, M_{par} >$,

Where M_{str} - model of structure, M_{par} – model of parameters.

$M_{str} = < M_{str}^1, \ldots, M_{str}^i, \ldots, M_{str}^n; P>$, M_{str}^i is the ith element of structure.
$P = \{P_1, \ldots, P_m\}$ – restricted set of relations defined on $M_{str}^1, \ldots, M_{str}^i, \ldots, M_{str}^n$,

$M_{par} = \{M_{par}^1, \ldots, M_{par}^i, \ldots, M_{par}^n\}$, M_{par}^i is the ith set parameters
$M_{par}^i = \{Par_1, Par_2, \ldots, Par_j, \ldots, Par_k\}$; Par_j is jth parameter of model.
We define critical parameters (or indicators) as:

$M_{par}^{cr} = \{Par_1, Par_2, \ldots, Par_i, \ldots, Par_l\}$; 'l' is maximum number of parameters. 'l' number depends from a design object.

$Mpar^{cr}$ - is a set of critical parameters - indicators, which affect possibility of assembly.

4.1.2 Design Mismatches

A described earlier, the main goals of implementation of IDMC are to detect and to resolve design mismatches.

In this case we should define what design mismatches are:

Definition 1

Design mismatches are inconsistencies between design goals G_i and the current design project $M_1(T)$, where T is time of design.

Obviously, the goals of design are a set of parameters (for design project) and predefined restrictions for these parameters. We propose that concurrency attributes are basis for definition of restrictions for parameters and structure of design project.

Definition 2

Design mismatches at the detail stage of design are inconsistencies between restriction parameters defined according to concurrency attributes and current parameters and/or structure of current design project $M_1(T)$.

4.1.3 Types of mismatches

We can define two main types of mismatches, which will be necessary to detect and resolve:

1. *Mismatches of integration.*
2. *Concurrency mismatches*

Mismatches of integration

There are assembly mismatches and necessary to detect and resolve these mismatches first, because they affect the design project integration.

Concurrency mismatches

Concurrency mismatches are mismatches of manufacturability, costability, manability, etc. (see section 2.1.1) related to concurrency attributes.

$Atr = \{Atr_1, Atr_2, Atr_3, \ldots, Atr_s\}$, where s – number of possible attributes.

for each field of implementation there may be different mismatches priorities and classes of concurrency attributes. This is because for one project the main priority is to provide a design for manufacturability, where another for the main priority is design for safety, for another – design for corrosion control, etc.
We will define Tax as taxonomy of design mismatches:

$Tax = Tax^{int} \cup Tax^{Atr2} \cup Tax^{Atr3} \cup, \ldots, \cup Tax^{Atrt}$.

where Tax^{int} - taxonomy of mismatches of integration (assembly mismatches); Tax^{Atri} is taxonomy of design mismatches related to ith concurrency attribute; t - is number of attributes for consideration in current project and $t \leq p$. A taxonomy of design mismatches is outlined in details in section 4.3.

4.1.4 Model of knowledge

IDMC uses a concept of distributed artificial intelligence - agents. In this case agents are represented as "virtual designers" who have internal abilities to receive information, to identify design mismatches and to prepare advice for the designer to find the best modification to resolve the mismatch.

Design knowledge model M_2 is used as a personal assistant for the designers D in design team Dt and helps to detect design mismatches and find the best modifications required.

The design mismatches are detected using a vocabulary of indicators and a taxonomy of design mismatches and resolved using a model M_2 - distributed model of designer's knowledge. We have:

$M_2 = \{A_1, A_2, \ldots, A_i, \ldots, A_n\}$; where A_i- agent.

Each agent is represented as part of an assembly and has knowledge about assembly part's geometrical configuration (structure) and concurrency attributes.

$A_i = \{W_1, \ldots, W_i, \ldots, W_n\}$, W_i is i^{th} Designer World

Each agent is represented as single knowledge-base which contains a set of Worlds of designer [Akman et al, 1990; Akman et al, 1987] to provide a knowledge about different aspects of a design project and concurrency attributes. Each designer world is represented as:

$W_i = \{K(M_{str}), K(M_{par}), K(Res), K(Indicators)\}$, where

K- knowledge, Res- design restrictions, I-indicators.

Another key ability of agents implemented for IDMC is their ability to adapt (to learn) using current information from designers to detect and resolve of mismatches.

If is clear, that during design time a W_i must be changed according to designer's knowledge and inter agent's communications.

Let T – time of design $\in t_1, \ldots, t_o$, when we have M_2 during a design time:

$M_2 (T) = W_1(t_1), \ldots, W_1(t_i), \ldots, W_1(t_n); , \ldots, W_i(t_1), \ldots, W_i(t_i), \ldots, W_i(t_n); W_o(t_1), \ldots, W_o(t_i), \ldots, W_o(t_o)$

The modification of W_i is a general process of adaptation. We have an *adaptation* of internal design knowledge (agent's knowledge) and *modification* of design projects. *Adaptation* is divided into two types: (1) *internal adaptation* and (2) *external adaptation.*

Definition 3

Internal adaptation of model of knowledge about design project - M_2 is a self-adaptation of knowledge using internal knowledge, as the result of communication with designers and agents.

Definition 4

External adaptation of M_2 is adaptation knowledge under the supervision of other agents and/or designers

4.2 Mismatch Control process and types of support for designer

The mismatch control process includes two main actions:

- Collision detection and
- Resolution of mismatches.

Changes in the structure or parameters of the design project, during the mismatch resolution action, are refereed to as the *modification* process:

Definition 5

Modification of design project M_1 is modification of M_{str} and/or M_{par}.

We can define types of IDMCS in order to support mismatch control. Such systems can be categorised into three major classes:

(1) Interpreters. Systems designed for interpreting the design situations. They utilise logical derivation sequences of the simplest form.

(2) Advisory Systems. Systems with enlarged knowledge base concerning the object of design. They analyse the situation obtained as a result of geometrical modelling and perform "k" lookahead of the user's action. When working with this type of systems designers will be able to improve (or get rid of) the mismatches manually, using their interpretation of the design situation.

(3) Prescriptive Systems. Systems with a capacity for controlling the mismatch detection and correctness process. They are capable of carrying out a series of modelling experiments, by themselves, to try several system models and modify their structure or characteristics. In this type of systems semi-automatic and/or automatic mismatch control can be applied. The work described here falls under the second and third class of

systems. We can define the hypotheses that IDMC-approach can be employed to advantage to provide a Concurrent Engineering design process that is as be to detect most mismatches as early as possible. For this, the approach should incorporate the necessary number of concurrency attributes - Atr, goals of design. The knowledge model should have a list of mismatches; presented in taxonomy which indicate a collision.

Fig. 7 represents a detailed mismatch control process.

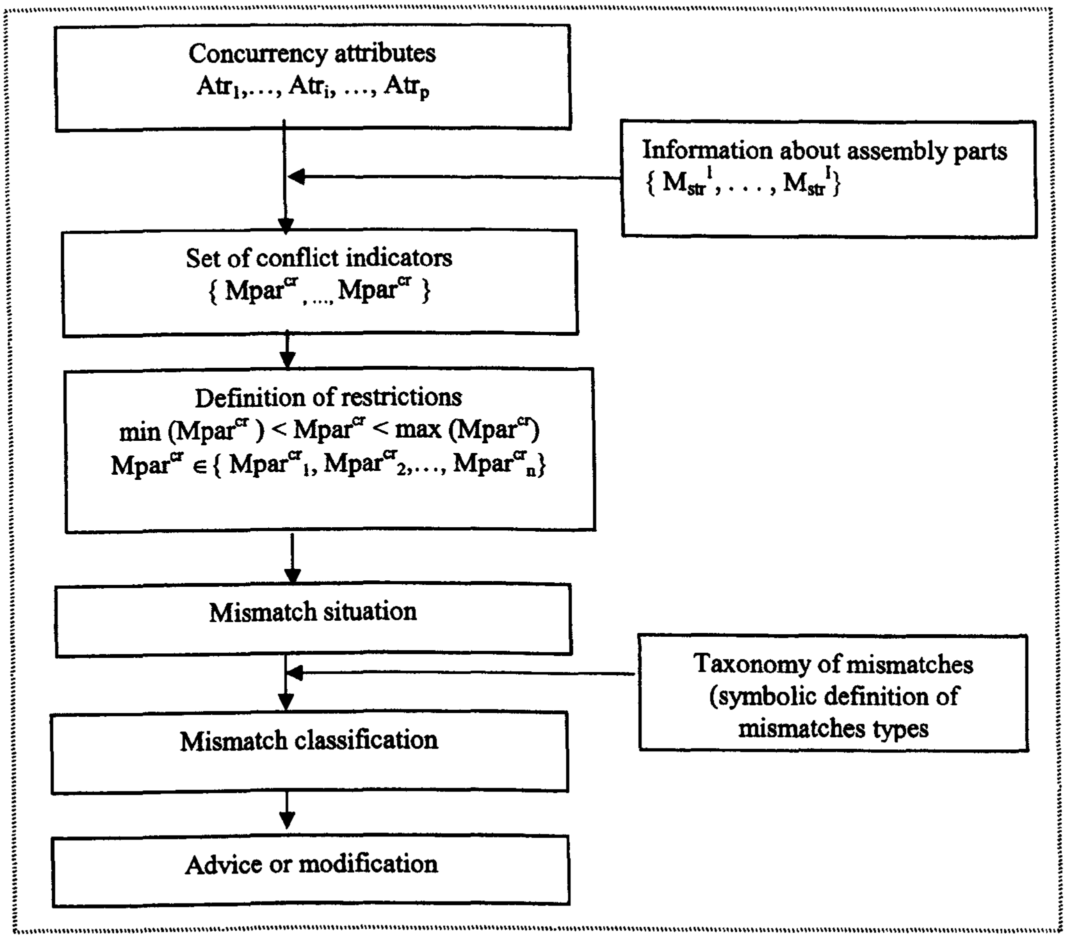

Concurrency attributes
$Atr_1, \ldots, Atr_i, \ldots, Atr_p$
Information about assembly parts
$\{ M_{str}^1, \ldots, M_{str}^l \}$
Set of conflict indicators
$\{ Mpar^{cr}, \ldots, Mpar^{cr} \}$
Definition of restrictions
$\min(Mpar^{cr}) < Mpar^{cr} < \max(Mpar^{cr})$
$Mpar^{cr} \in \{ Mpar^{cr}_1, Mpar^{cr}_2, \ldots, Mpar^{cr}_n \}$
Mismatch situation
Taxonomy of mismatches (symbolic definition of mismatches types
Mismatch classification
Advice or modification

Figure 7: Mismatch control process

As can be seen, the mismatch control process is included in a number of different stages in the distributed environment. The information about conflict indicators and the taxonomy is distributed between agents which can provide collaboration for definition of mismatch situation.

By examining these results it will be possible to build a general picture of the inter-relationships between tasks in Intelligent Distributed Mismatch Control Process. The basic scheme of IDMC is illustrated schematically in Fig. 8.

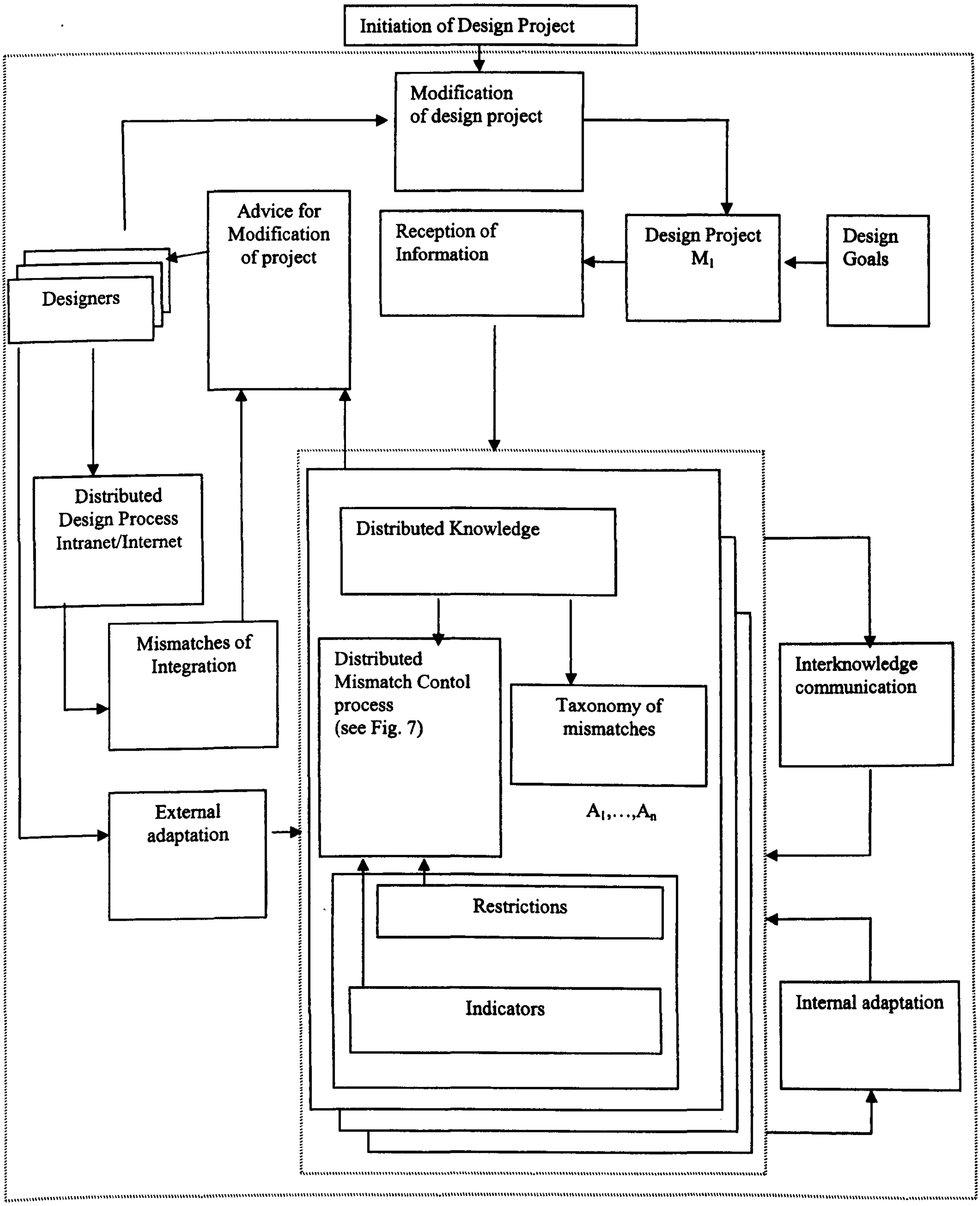

Initiation of Design Project
Modification of design project
Advice for Modification of project
Reception of Information
Design Project M_1
Design Goals
Designers
Distributed Design Process Intranet/Internet
Mismatches of Integration
External adaptation
Distributed Knowledge
Distributed Mismatch Contol process (see Fig. 7)
Taxonomy of mismatches
$A_1,\ldots,A_n$
Restrictions
Indicators
Interknowledge communication
Internal adaptation

Figure 8: Intelligent distributed mismatch control. Basic scheme

This diagram is intended to show how Mismatch Control Process must be organised in order to support design process. It is proposed that further, more in-depth analysis be performed to identify more relationships and properties of tasks and tools.

It can be seen a mismatch control includes three stages: receiving design project information, identification (classification) and generation of result. The vocabulary of indicators was used for identification of mismatches. This vocabulary includes a classification tree and indicators (taxonomy of design mismatches).

Of course, it is important to understand that this scheme provides a general structure, which will be necessary to investigate, and detail organisation will be described later. Based on information described above we present a general principle of Intelligent Distributed Mismatch Control as described below:

Table 5 General principles of Intelligent Distributed Mismatch Control

No.	**Description**
1	Define a number of elements of assemblies
2	Specify and priorities the concurrency attributes for this type of design
3	Identify relevant conflict indicators (critical parameters)
4	Identify appropriate groups of restrictions
5	Detect a mismatch situation using a intelligent agent communication
6	Using appropriate taxonomy of design mismatches, provide a mismatch classification.
7	Provide a advice for designer about possible modification of design project or provide automatic/semi-automatic modification

As will be shown later, each of these principles will need to be considered when creating an integrated Distributed Mismatch Control System for use for Design.

On next stages of IDMC research it is necessary to define:

- Taxonomy of design mismatches
- Mismatch Detection algorithms (collision detection), Resolution of mismatches
- External and Internal Adaptation Algorithms
- Knowledge-base organisation

It is important to realise that the above four items are not intended to provide an exhaustive list. Also many of these parts can and should be inter-related.

The IDMC-approach was implemented for the organisation of the initial multi-agent framework as a basis for mismatch control process.

4.3 Design Mismatches. Classification and Taxonomy

Important developments in this area include models reported by [Klein, 1991, 1995; Matta, 1997; Castelfranchi, 1996], but these taxonomies, in general, are more oriented towards the conceptual stage of design process.

We propose a conceptual framework for development of taxonomy for the detail stage of the design.

Firstly, we will classify design mismatch *according to the levels/types of information needed for its detection.*

We have:

- syntax level - ordinary geometric mismatches (size, diameter, geometric type, parts orientation,...)
- semantic level - *complex assembly mismatches* - analysis of geometric and materials characteristics for checking assembly possibility.
- pragmatic level - the complex mismatches are connected to design/concurrency attributes, such as mismatches of manufacturability, manability, costability, serviceability, etc.

Accordingly the types of design mismatches we were defined *mismatches of integration and concurrency mismatches (see Section 4.1.3).*

The general structure of development of taxonomy is needed to indentify the relations between design requirements, stages of the design process, design mismatches, fields of applications and critical parameters.

This strucure will be used for design of detailed taxonomy for applied field such as aerospace design, automotive, mechanical engineering.

The general overview of development taxonomy of mismatches is presented in Fig. 9.

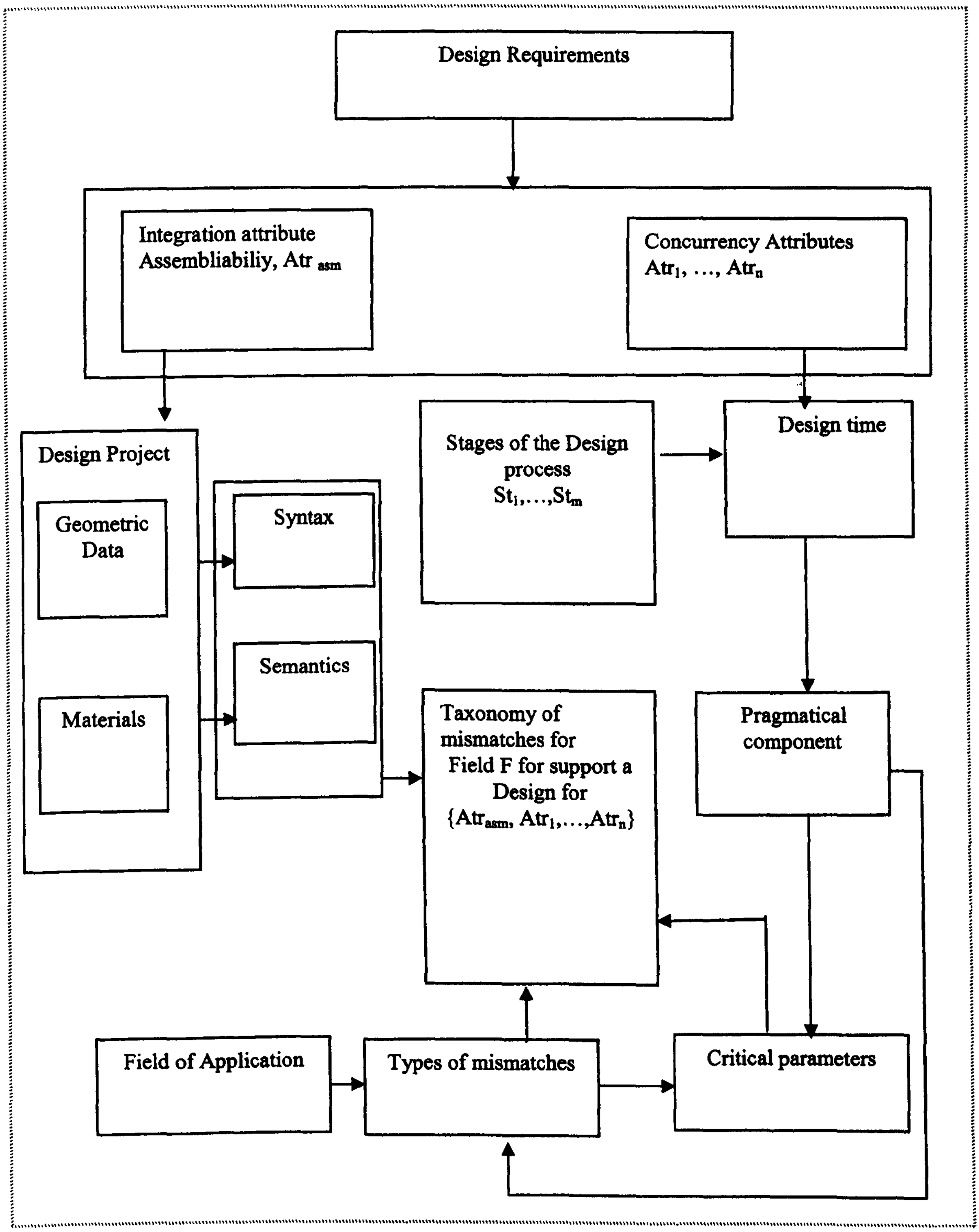

Design Requirements
Integration attribute Assembliabiliy, Atr $_{asm}$
Concurrency Attributes $Atr_1, \ldots, Atr_n$
Design Project
Geometric Data
Materials
Syntax
Semantics
Stages of the Design process $St_1, \ldots, St_m$
Design time
Taxonomy of mismatches for Field F for support a Design for $\{Atr_{asm}, Atr_1, \ldots, Atr_n\}$
Pragmatical component
Field of Application
Types of mismatches
Critical parameters

Figure 9: A Development of Taxonomy of Design Mismatches

Using the proposed model of taxonomy development it is possible to define a practical taxonomy, given a known field of implementation – aerospace, automotive, electronics. The development of taxonomy is complicated and long process which included a definition of critical parameters, indicators, restrictions and attributes and based on knowledge engineering component, as interviews and questionnaires (Appendix B).

It is clear, that definition of complex taxonomy of mismatches is a very important part of research in the field of general design methodology. The definition of complex taxonomy of design mismatches will help to solve three main theoretical problems:

- to identify the scope of computer tools for design,
- to identify similarities between different domains, and solve a practical problem:
- to develop a tools for automatic/semi-automatic classification of mismatches.

The general overview of taxonomy of mismatches is presented in Fig. 10.

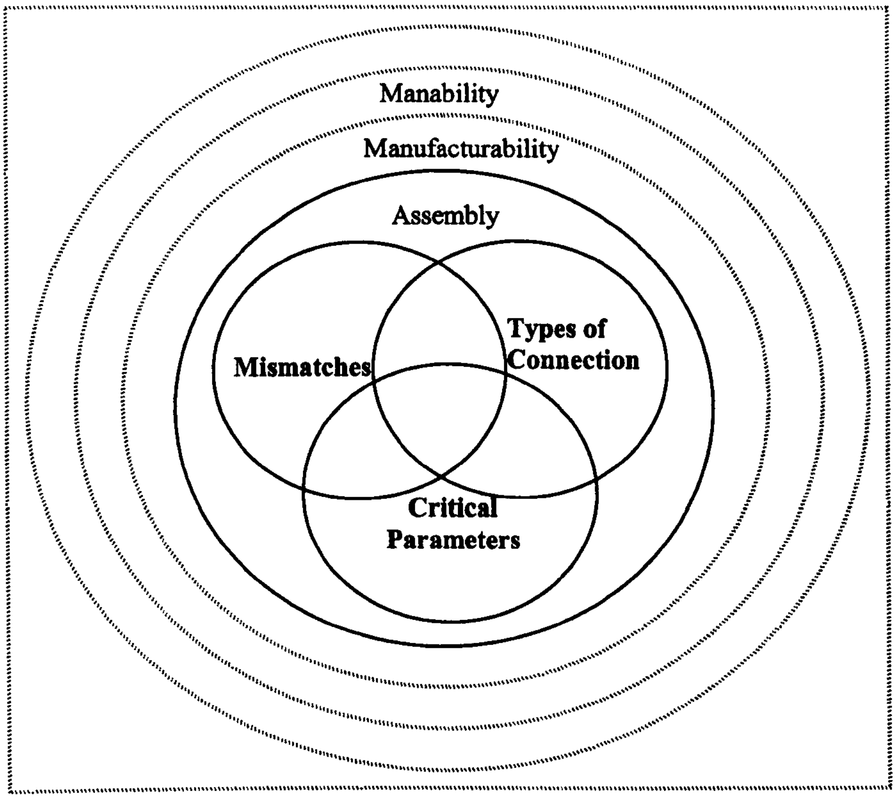


Figure 10: A Taxonomy of Design Mismatches. General Overview

The problem of devising a fully-fledged taxonomy for design mismatches is a very complex one. As described earlier, the design is a multi-disciplinary task that involves several stages. These stages include input data, conceptual design, basic structural design, detail design, production design, manufacturing processes analysis, and documentation (see [McMahon & Browne, 1993]).

A broad classification based on geometrical mismatches is presented in [Bechkoum, 1997]. Some of the important parameters to consider in the case of Design for Assembly (DFA), are presented in [Lee et al., 1993].

Our taxonomy uses some of these known parameters, but is especially oriented for implementation for mismatch detection during the integration phase of mechanical engineering design. In this case in terms of concurrency attributes our taxonomy provides supports design for assembly [Lefever and Wood, 1996], disassembly, and manufacturability.

The implementation of taxonomy will be a typological extrapolation for organisation of distributed knowledge base for organisation of automatic classification mismatches as internal agent's ability.

The next chapter describes a formal description of structure of multi-agent framework and dynamics of multi-agent framework.

4.4 Conclusion

This part has focused on the definition of a Methodology for Intelligent Distributed Mismatch Control – IDMC-approach.

Intelligent Distributed Mismatch Control was described in details. The model of design project, definition of design mismatches and types of mismatches and Model of knowledge about consistency checking process are introduced.

The Mismatch Control process including negotiation, detection of mismatches and handling of mismatches is represented. Also, the types of IDMCS in order to support mismatch control are described. Such systems have been categorised into three major classes: (1) Interpreters; (2) Advisory Systems; (3) Prescriptive Systems.

The new framework for definition of taxonomy of design mismatches is outlined. Using the proposed model of taxonomy development it is possible to define a practical taxonomy, given a known field of implementation - i.e. aerospace, automotive, electronics, and numbers of concurrency attributes.

The IDMC-approach will be a solid basis for development a distributed problem solving models for engineering design.

The next chapter described the application of IDMC-approach for agent-based design environment.

Chapter 5 The application of the Methodology. The Multi-agent Framework

5.1 Formal description of the multi-agent framework

5.1.1 The structure of multi-agent framework

We described earlier a general model of IDMC-approach and in this chapter we provide a detailed description of the structure and its definition of elements needed for mismatch control support.

As can be seen, the main problems which will considerate within framework is distributed knowledge-base organisation and agent's types, and cooperation during design process.

The conceptual framework of the IDMCS is shown in Fig. 11. The framework assumes that the design knowledge is encapsulated within the different members of agent community.

Conceptual framework (CF) may be presented formally as follows:

$CF = \{AP_1, \dots, AP_t, \dots, AP_n\}$,

AP_t is the t^{th} Assembly Part, $t = 1,2, \dots, n$.

$AP=\{DA_1, \dots, DA_i, \dots, DA_m, CA_1, \dots, CA_j, \dots, CA_k\}$,

We define two types of agents for this framework:

DA_i is the i^{th} Design Agent (D-agent), $i = 1, 2, \dots, m$.

CA_j is the j^{th} Control Agent (C-agent), $j = 1, 2, \dots, k$.

Each DA_i consists of eight elements: *FB* - facts base, which includes information about geometric characteristics of the part and material type. *KB* - knowledge-base. *K* - corrector block - which adapts knowledge base, as a result of communications with any other agents. *I* - inference engine. *LI* - local interface mechanism.

Each CA_j consists of: MB - metaknowledge base, knowledge-base of control agent, inference engine, corrector block, local and global interface mechanism (GI).

GI, LI are provided transfer data between agents. Because agents are using a different agents communication languages (ACL), global and local interface are translated a messages from external ACL to internal description and from internal description to external representation. K- corrector is realised for internal adaptation of knowledge and fact bases that will be described later in this report.

5.1.2 Organisation of agents

In general, the design and control agents consist of two types of knowledge. In facts–bases data about current research project as are represented by frames. Knowledge base, as active warehouse of knowledge about methods of agent's collaboration for conflict resolutions are represented by rules.

In our case, as described earlier (section 4.1.5), conflict between agents is indicated by mismatches of integration (level of D-agents) or mismatches of concurrency (level of C-agents). The organisation of distributed knowledge-base is described in detail in section 4.5, and general structure of C and D-agent is represented in figure 12.

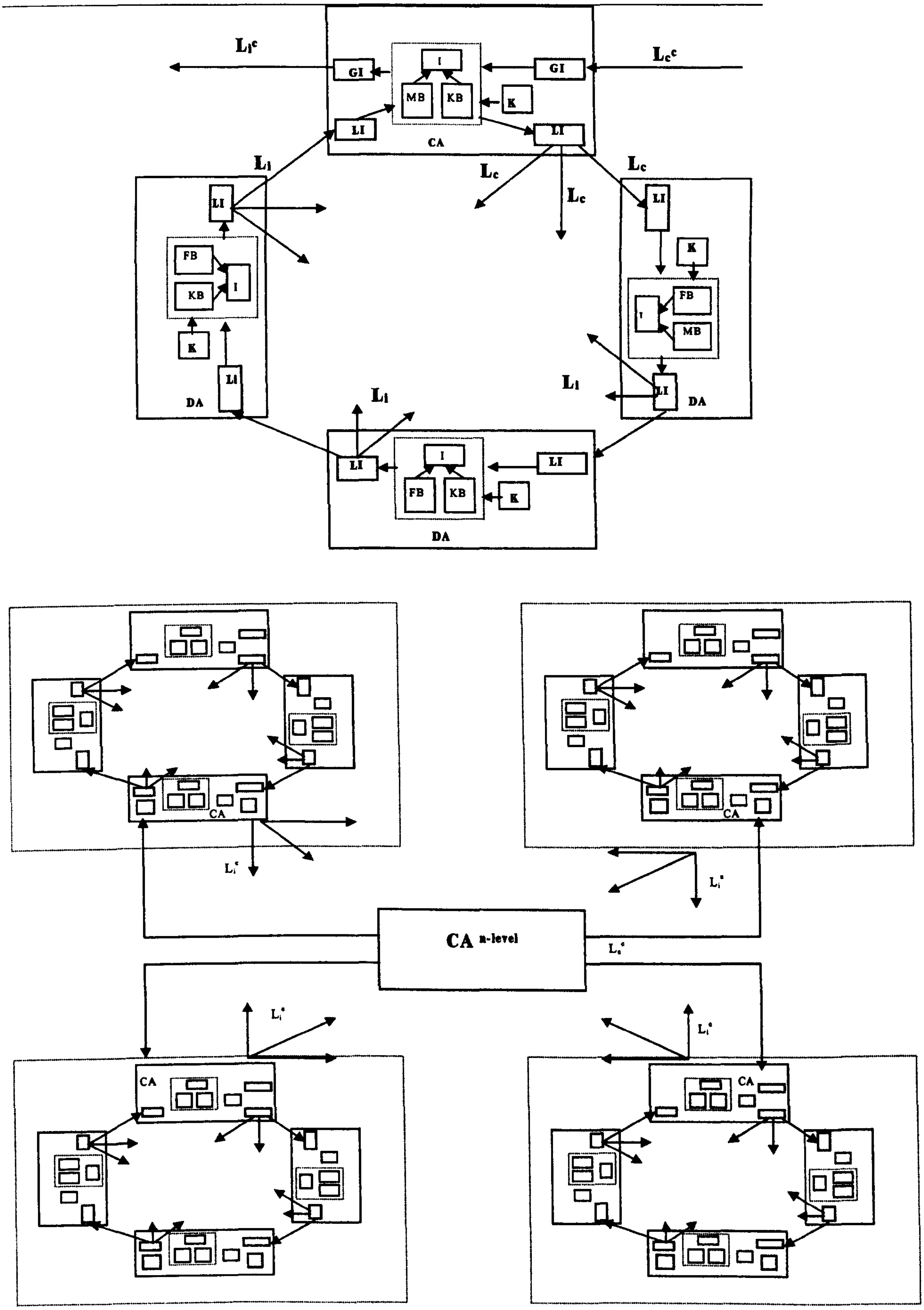


Figure 11: The architecture of Multi-agent Framework

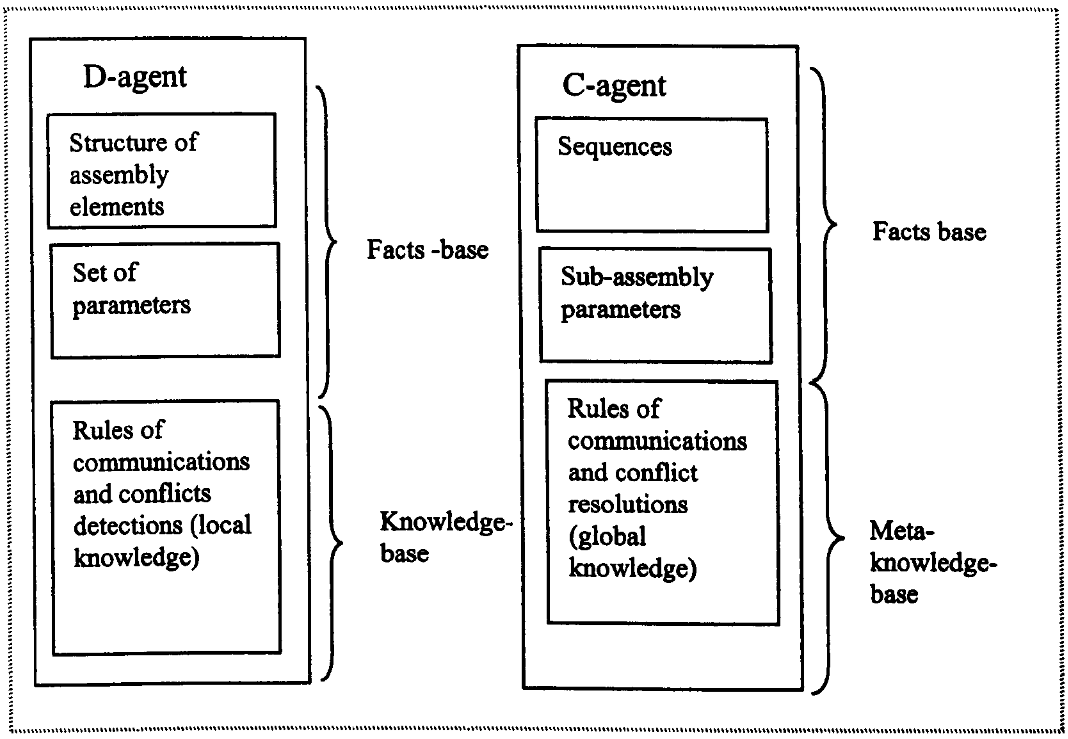

D-agent
Structure of assembly elements
Set of parameters
Rules of communications and conflicts detections (local knowledge)
Facts -base
Knowledge-base
C-agent
Sequences
Sub-assembly parameters
Rules of communications and conflict resolutions (global knowledge)
Facts base
Meta-knowledge-base

Figure 12: Structure of D- and C-agents

We represent conceptual framework as community of schedule [Liu and Sycara, 1994] and reactive agents [Brooks, 1990; Kaelbling et al, 1995; Kaelbling and Rosenschein, 1990]. In our case D-agent is a reactive agent, which negotiate with other D-agents using design' schedule (assembly sequence) generating C-agent.

A reactive agent is an entity that may be represented by an independent program that knows everything about itself including its relationships with other agents. The principle of emergence states that intelligence in reactive agents emerges from interaction of agents among themselves and with their environment. The principle of situatedness states that intelligence of a reactive agent is situated in the world and not in any formal model of the world build in the agent [Brooks, 1990].

5.1.3 Communication among agents. Communication Protocol (Language - L2)

It is clear that communication – ability to prepare, to send and to receive messages is critical for of multi-agents systems. The proposed communication protocol (CP) for the above agents for IDMC support is as follows:

$$CP=\{L_{c1}, \dots, L_{cm}; L_{i1}, \dots, L_{im}; L_c{}^c{}_1, \dots, L_c{}^c{}_k; L_i{}^c{}_1, \dots, L_i{}^c{}_k\},$$

where:

L_i - information language, which describes current situation into multi-agent system,

L_c - control language, which includes imperative commands about adaptation fact base of design agent for mismatch improvement, adaptation and modification D-agent's knowledge-base.

$L_i{}^c$ - information language, which describes current situation for control agents,

$L^c{}_c$ - control language, which adapt meta- and knowledge base of C-agents.

The definition of languages of communication depends on the types of agent's communication languages (KQML, FIPA) as described earlier (see 2.2.3) and analysed in [Woldridge, 2000]. In this research we are using a FIPA standard for ACL to provide a hub for realisation IDMCS using a ZEUS toolkit.

5.2 A multi-agent Framework: Learning, Cooperation and Negotiation. Dynamics in multi-agent Framework

5.2.1 Theoretical framework for language L3 type

Techniques for multi-agent systems representation are include: Petri Nets [Ferber, 1999], automata theory [Kim, 1989], schemata [Holland, 1968], algebraic/language-based models [Gorodetski and Lebedev, 1998], and logic models. To define a multi-agent framework dynamics we are using automata theory [Hopcroft and Ullman, 1979] as a formal basis.

Automata theory investigates fundamental principles shared by artifacts such computers and control systems [Kim, 1989], as well as natural systems such as human nervous system. Automata theory will be used as a theoretical foundation toward a unifying framework for IDMCS.

The traditional perspective of automata theory is characterised by a focus on information processing issue [Hopcroft & Ullman, 1979]. We have input information, computation block-automata, and results - outputs. Our agents can be viewed as a learning automaton and analysis situation in multi-agent framework.

The implementation of finite-state automata is interesting for multi-agent systems modelling because, according to Ferber [Ferber, 1999] it is easy to describe the behavior of an agent capable of memorising the state in which it finds itself and, secondly, the concept is backed up by great deal of theoretical support and has been used in many computing fields such as computer architectures, formal languages, networks, and etc. There are some results implemented of automata approach for MAS [Kaelbling, 1995; Kaelbling, 1990] named as situational automata. The book [Duffy and Andreasen, 1999] described the implementation of the Petri-nets formalism (in general extension of automata approach).

It is also clear that automata theory has some restrictions as limited number of states. Using automata theory is possible to describe only sequential processes, but in our case this approach is acceptable on stage of conceptual describing of possible communication within multi-agent framework.

We are using automata approach as a basic investigation of models of adaptation and mismatch resolution within the multi-agent framework.

We have automata network:

$M_2 = \{ A_{CA}^1, \ . \ . \ . \ , A_{CA}^n , A_{DA}^1, \ . \ . \ . \ , A_{DA}^m \}$,

where

n- number of C-agents,

m - number of D-agents.

Let us define the two types of automata:

$A_{CA} = (X^{M1}, X^{DA}, Y^{CA}, Y^{DA}, Z, F, Q, z_o, z_k)$,

$A_{DA} = (X^{DA}, X^{CA}, Y^{DA}, Y^{CA}, Z, F, Q, z_o, z_k)$.

Where:

X^{DA}- set of input information from design project M_1

X^{CA}- set of input information received from other D-agents

Y^{CA} – set of output information sent to C-agents

Y^{DA} – set of output information sent to D-agents

$$\left.\begin{array}{l} F{:}Z^*X \Rightarrow Z \text{ - } A_{DA} \text{ transition function} \\ Q{:}Z^*X \Rightarrow Y \text{ - } A_{DA} \text{ output function} \\ z_0 \text{ - initial state} \\ z_k \text{ - final state} \end{array}\right\} \textbf{D-automata}$$

X^{M1}- set of input information from design project M_1

X^{DA}- set of input information received from other design agents

Y^{CA} – set of output information sent to C-agents

Y^{DA} – set of output information sent to D-agents

$F:Z*X \Rightarrow Z$ - A_{CA} transition function

$Q:Z*X \Rightarrow Y$ - A_{CA} output function

z_0 - initial state

z_k - final state

C-automata

Using an automaton approach it will be possible to analyse a communication strategies in multi-agent network and to develop mismatch detection and resolution scheme.

5.2.2 Communication and conflict resolution

For multi-agent systems with IDMC abilities the process of communication is critical because during this process the system detects mismatches.

We can state that in order to resolve the conflicts in multi-agent cooperation we will use an arbitration scheme. Arbitration from C-agent will stop disagreement between D-agents when conflict situation is presented and modification of project (fact base of D-agent) required. We can define the approach as co-ordinated collaboration [Duffy & Andreasen, 1999] when we have compatible goals of agents, insufficient resources and insufficient skills.

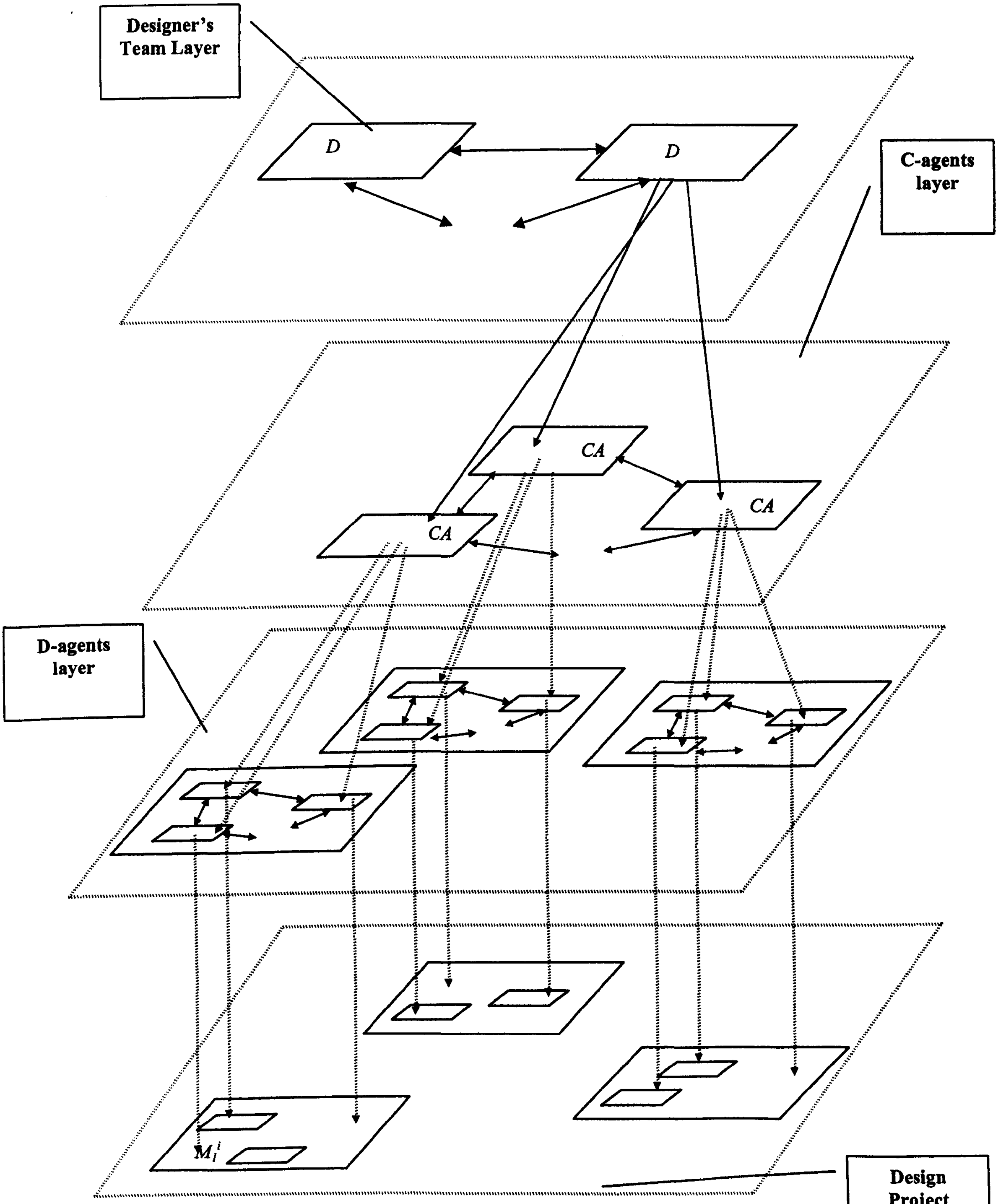


Figure 13: IDMC representation: Dynamic automata network

In our case goals are assembly possibilities, resources are sets of parameters, and skills are mismatch detection abilities.

In our multi-agent framework we have three levels of vertical communications: (1) reactive level (D-agents- Design project), (2) control level (C-agents -designer), (3) designer level and two levels of horizontal communication as (1*) between D-agents and (2*) between C-agents. (Fig. 14).

We are using general two-layer architecture as described earlier, with D-and C-agents, and coordination and combination of groups of agents is realised by Designers using a assembly sequences.

Figure 14 shows adaptation scheme in Multi-agent framework. The different layers are described below:

Reactive layer (Vertical) (Figure 14a):

Each D-agent operates as an independent entity and interacts asynchronously with associated assembly parts.

The communication between D-agents and the associated parts of the design project is a process of elimination of inconsistencies.

These inconsistencies may be the result of a modification of the Design Project or a self-adaptation of the knowledge base of D-agents.

Control layer (Vertical) (Figure 14b):

The communication between C-agents and associated D-agents is a process of elimination of inconsistencies between assembly parts, when D-agents are unable to resolve it, using internal knowledge and/or horizontal communication.

This is client-server communication (under the supervision of the C-agent). C-agent receives the new information from D-agents using syntax of L_i. The result is external adaptation of knowledge-base of D-agents (using syntax of L_c), according to C-agents meta-knowledge base information, if mismatches occur.

Designer layer (Vertical) (Figure 14c):

The communication between C-agents and designers (design team) is a process of elimination of inconsistencies between assembly parts, when C-agents are unable to resolve it, using internal knowledge and/or horizontal communication.

This is human-computer communication. The result of communication is external adaptation of metaknowledge-base of C-agents.

Horizontal Communication:

The Horizontal communication takes the form of a negotiation between associated D-agents (Figure 14d) (D-agent to D- agent relation); C-agent to C-agent (Figure 14e), and Designer to Designer (Figure 14f).

The communication is at peer-to-peer level. D-agent negotiate with other D-agents, using L_i. C-agent negotiate with other C-agents, using L_i^c and L_c^c. Each of these communications aim to eliminate mismatches.

The communication between designers is normally organised using e-mail, telephone and other forms of communication because it is necessary to the effective organisation of the design team.

The optimal organisation of communication for design team is a very important part of the design process, but it is outside the scope of this research.

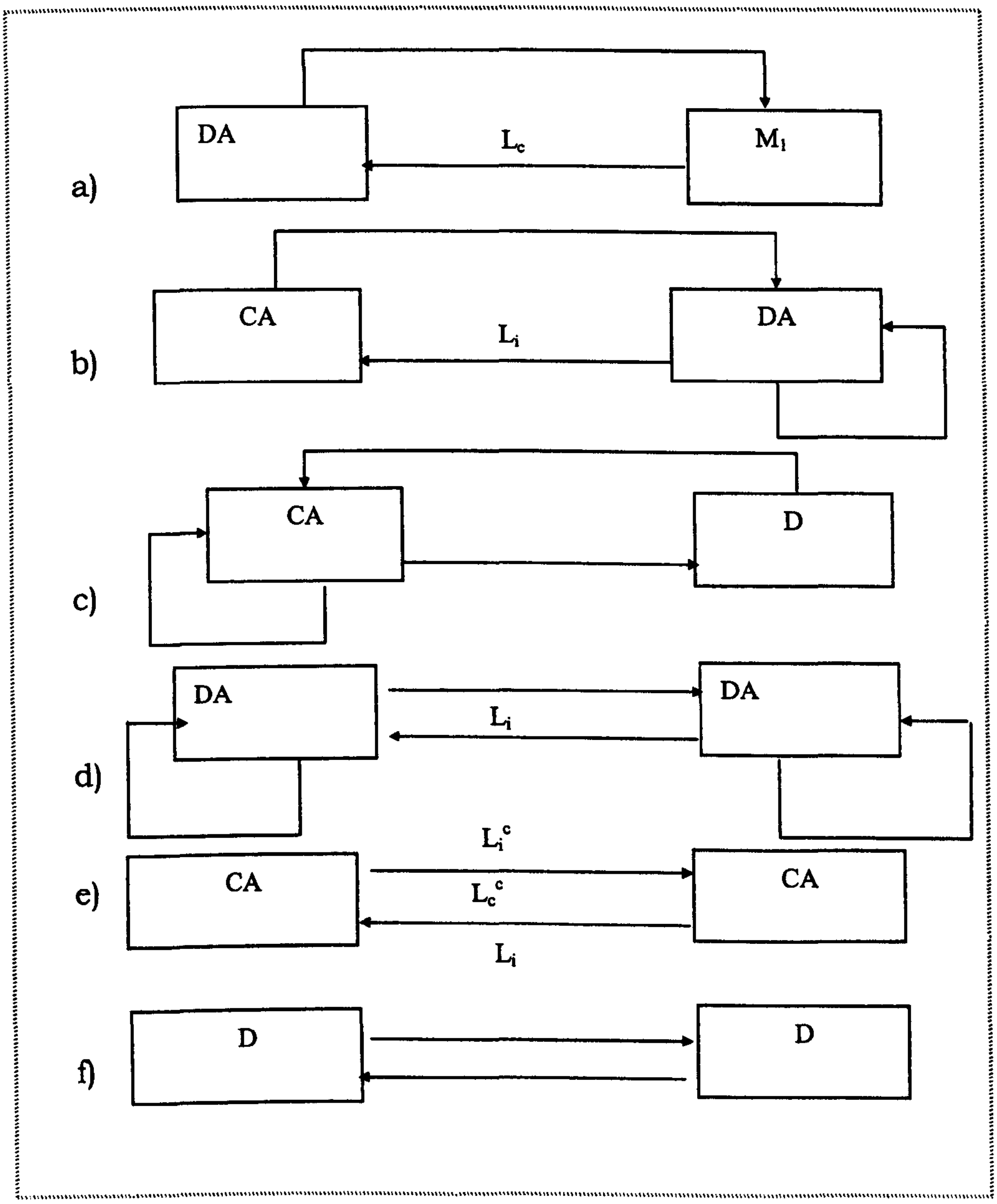


Figure 14: Communication in Multi-agent Framework

The next figure (Fig. 15) shows an adaptation mechanism of C- and D-agents.

It is clear that the process of adaptation (we can say learning) in a multi-agent environment is complicated because the environment changes as other agents learn. At the present time, researchers have developed different models of agents learning, using different mathematical and other approaches, such as Bayesian networks, neural networks, economic bargaining negotiation model, Q-learning and others [Rocha, 1999].

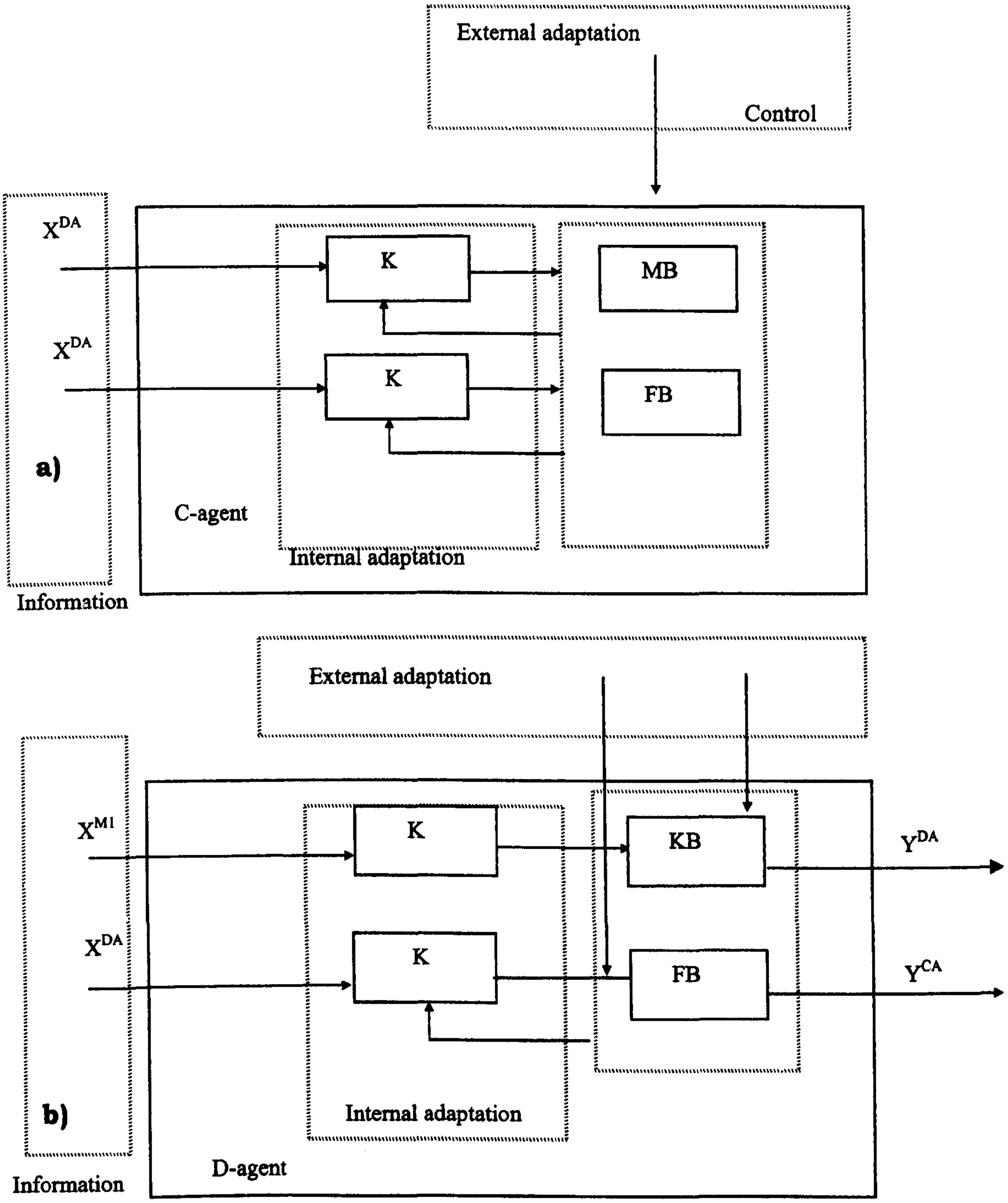


Figure 15: Adaptation mechanism of C-and D-agents

In fact, we are using an automata approach for formalisation of IDMC-process and the organisation agents as learning automata will be give us a possibility to provide an adaptation during design process.

To define a best algorithm for learning is important and will be investigated at the stage of creation of extended prototype of IDMC.. In IDMC process of mismatch resolution is strongly connected with agent negotiation for mismatch detection and adaptation of fact-base and knowledge-base of C- and D- agents.

The next section represents possible external and internal adaptations mechanisms for conflict (mismatch) resolution.

5.2.2.1 External adaptation

As can be seen, (Fig. 15) the external adaptation is a direct change of Fact and Knowledge-bases of C- and D-agents. For C-agents the adaptation of KB is provided by designers in the event if they do not satisfy the results. Designers or C-agents provide the external adaptation. C-agents are using an internal effector for providing an adaptation. Of course, the definition of effectors for C-agents is still not very clear, especially in case of implementation of Design process. The results of external adaptation provided by C-agents, in case of mismatch resolution, may not be satisfied for designers. In this case, the designers should have a mechanism to stop a prescriptive capability of IDMC and to provide a mismatch resolution manually, according to IDMCS advice.

The internal adaptation of distributed knowledge-base is described in the next section.

5.2.2.2 Internal adaptation

We propose an algorithm for internal adaptation of knowledge-base similar to 'classifier' based systems originally introduced by Holland [Holland, 1968].

According to [Duffy and Andreasen, 1999] classifier-based systems are based on a variation of production rules. Using a credit attribution system - rewards the rules that have given rise to a 'good' action, that is, an action considered made to arrive a goals. The weight of these rules is increased; whereas in the opposite case rules that have not brought any benefit to the agent have their weighting reduced.

Another important characteristic of 'classifier' based system is that system reproduces new rules using genetic algorithms (using mutation and cross-overs).

We are proposing a hypothesis that 'classifier'-based algorithms will be more acceptable for the design mismatch process (opposed to clear productions systems, connectionist architecture, and other approaches). This is because, normally, the design system for support of detailed design stage, is not restricted by a set of acceptable parameters, restrictions of structure and this approach will give an additional mechanism for verification of distributed knowledge base. Before real design will be started, the designer will analyse an advice generated by IDMCS, provide a verification of results and change the weights of productions.

At this stage we are not using genetic algorithms to generate new rules and provide an adaptation on restricted numbers of rules.

The proposed scheme needs a special organisation of knowledge-base. In our case an initial algorithm of adaptation of knowledge-base is based on penalty–learning strategies. The special form of rules is:

Q_1; Q_2; P $A \rightarrow B$, N,

Where Q_1 - design stage, Q_2 - model world of designer, P-condition, $A \rightarrow B$ - traditional kernel of rule (IF A THEN B), N – postcondition

Define:

i - set of rules, j – set of alternative rules, i = 1, ..., n, M_{ij} – rule, M^+ - active rule, M^- - passive rule. The active rules are rules with Q_2 =max and P=const. Define a knowledge-base structure (Fig. 16):

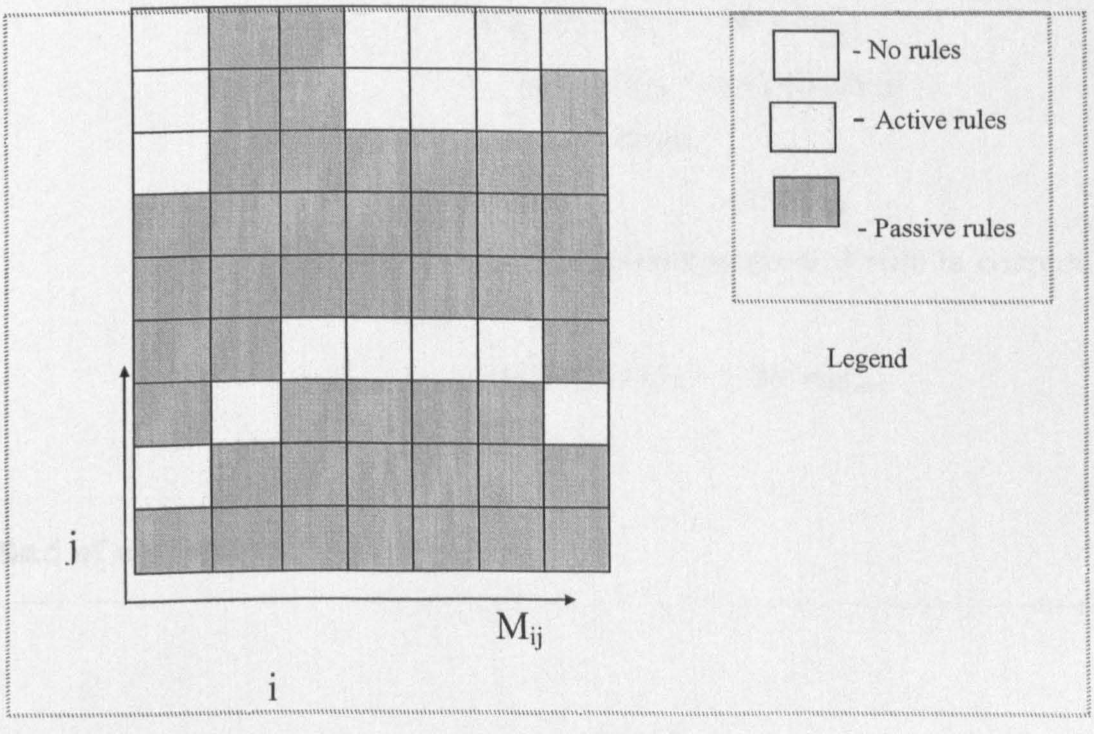


Figure 16: The structure of knowledge-base

We have a proposal for adaptation algorithm:

```
Begin of algorithm

Define the design stage $Q_1$
Analysis of situation using P.
Define $Q_2$ = max.
    Rules execution
            If check N implementation of rule is fail
                then
                    $M_{det}\ Q_2 = Q_2 - 1;\ M^- = M_{det}$
                        if $Q_2.M_{i,j} = \max(Q_2.Mij)$
                            then
                                $M^+ = M_{ij};$
                ELSE check N implementation of rule is correct
                Then
                    $M_{det}\ Q_2 = Q_2 + 1;\ M^+ = M_{det}$

End of algorithm
```

As can be seen the external adaptation of design project is provided by changing a weight of rules according to the condition N analysis result.

The implementation of this algorithm is an interesting step towards a more comprehensive solution but is far from being defects free. For example, the definition of classes of active and passive rules is needed for the participation of the designer in the initial stage of design. The algorithm adapts the knowledge-base, but does not suggest a correctness of implementation of rule and does not have a mechanism to add additional rules in distributed knowledge-base.

In this case, the implementation of this algorithm may require the use of genetic algorithms and neural networks, as the best way forward, for automatic adaptation of knowledge-base of C- and D-agents.

5.3 The distributed knowledge-base. Definition of language for presentation of knowledge (L4)

MB, KB of C-agents and KB of D-agents are represented using production rules called Receptors for carrying out an analysis of the design situation; production rules called Classifiers for classification of situations according to the necessity of control actions; and production rules called Effectors.

Rules in KB and MB of D- and C-agents provide analysis of design situation using experts knowledge. Each rule M_p, $p = 1, \dots, r$, is characterised by premise part, comprising the IF preconditional statements, and the consequent part (THEN part), comprising the inferred outputs.

Receptors are represented as:

IF < situation = ST > THEN < start C >,

where ST - set of mismatch situations, $ST \in \{ST_1, ST_2, \dots, ST_p\}$,
C - classifier. Classifiers are divided into three types:

IF < ST > THEN < estimation >,
IF < ST > THEN < estimation and recommendation>,
IF < ST > THEN < start E >,
where E - is effectors. Effectors are divided into 2 sets:

$E = < E_{ext}, E_{int} >$,

E_{ext} is a set of rules for modification and external adaptation of the FB of D-agents and E_{int} - internal adaptation MB and KB of C-agents.

The distributed knowledge-base uses a frame-based representation of the facts.

FB of D-agent including frame facts containing information about parts geometric and material consistency

Tree types of frames for D-agent have been investigated:

FMstr = (orientation (Angular, Perpendicular, Parallel), geometric coordinates (X,Y,Z),..., Size of (entity, parts, ...), Type of Connections, (Type of screw(Capscrew, Setscrew,...,), ...,),...)

FMpar = (material data (Deformation, Strength, Hardness, Stress,...), glue type, material number,.., ...)

For C-agent:

CF = (Number of Parts of subassembly, assembly sequence, relations between levels of design hierarchy).

The full initial typology of knowledge-base organisation is represented in Appendix A, and summary of distributed knowledge-base organisation is presented in Table 6.

Table 6: Distributed-Knowledge base organisation

Notation	**Name**	**Type of Agent**	**Part of ..**	**Knowledge representation paradigm**
G	Design goals	C-agent	Meta-knowledge-base	Rules
Ars	Assembly Restrictions	C-agent	Meta-knowledge-base	Rules
T	Matherial type	D-agent	Fact-base	Frame M_{par}
Po	Orientation Information	D-agent	Fact-base	Frame M_{str}
Ch	Geometric data	D-agent	Facts Base	Frame Mstr
Tax	Taxonomy of design	C-agent	Fact base	Frame CF

Notation	**Name**	**Type of Agent**	**Part of ..**	**Knowledge representation paradigm**
	mismatches			
Re	Restrictions	C and D-agent	Knowledge-base	Rules
R_1, R_2 ,R_3	Different types of receptors	C-and D-agent	Knowledge-base	Rules
C	Classificators	C and D agent	Knowledge-base of D-agent and metaknowledge-base of C-agent	Rules
As	Assembly sequence	C-agent	Fact-base	Frame CF
E	Different types of effectors	C-agent	Knowledge-base	Rules

This proposal for organisation the distributed knowledge base presents a basic skeleton for future extensions, which will be necessary in order to provide, developed mismatch support capability. For instance, for design safety and aesthetics we should include a colors and shape characteristics, for design for corrosion control [Banis et al., 2000] – drainage location, sealant types, galvanic coupling of materials, etc. Another important characteristic of distributed knowledge-base of IDMCS is to provide support for a hierarchical structure of assemblies and sub-assemblies.

The initial information about hierarchy of design is saved in frame of fact-base of C-agent (frame- CF) and will give an access to different levels of assembly.

5.4 Conclusion

The IDMC approach focused on theoretic and conceptual issues of Intelligent Mismatch Control and gave some insight into current strategies for its organisation of multi-agent framework of IDMCS. We represented conceptual framework as community of and reactive agents. In our case Design agent is a reactive agent, which negotiate with other design agents using design' schedule generating by control agent.

The agent communication protocol (CP) based on formal notations was described.

The possible external and internal adaptations mechanisms for conflict (mismatch) resolution have presented. The Learning, Cooperation and Negotiation within multi-agent Framework have analysed.

The structure of meta- and knowledge-base of design and control agents is represented using production rules called Receptors for carrying out an analysis of the design situation; production rules called Classifiers for classification of situations according to the necessity of control actions; and production rules called Effectors.

This proposal for the organisation of the distributed knowledge base presents a basic skeleton for future extensions, which will be necessary in order to provide developed mismatch support capability.

This formal model provides a foundation on which this research is based, providing experience in research process as well as illuminating some interesting areas, which inform my subsequent research. The next chapter describes the organisation of IDMCS and the implementation IDMC for aerospace design applications.

Chapter 6 The Principles of Organisation of Intelligent Distributed Mismatch Control System (IDMCS). System's Development and Overview

6.1 The main principles of IDMCS development

In this chapter the main principles of IDMCS development based to IDMC-approach, including a definition of stages necessary for organisation of IDMCS are defined.

In many areas software engineering methods have been developed. Multi-agent approach is a new paradigm of software organisation and new methods of design and such sort products are needed. In research conducted by Brasier [Brasier et al., 1989] the general methodology of MAS developmment is described. In this chapter we will try to identify the main differences between development of MAS and IDMCS.

To develop the IDMCS we need three main stages such as initial definition (conceptual stage), detail stage and technological stage.

At the initial definition stage the knowledge engineering issues are defined, types of mismatches needed to be resolved, using taxonomy of design mismatches.

Detail design stage

At this stage the negotiation process methods, type and dimension of multi-agent frameworks, agent negotiation algorithm, set of CA and DA agents, mismatch detection and resolution algorithm are defined.

The technological stage

Technological stage is more about how to implement the theoretical framework for a real life problem and implementation of IDMCS for a specific industrial application: Industry sector, field, e.g. Aerospace, automotive, communications, defense.

The technological stage depends on the field of IDMCS implementation. The taxonomy of mismatches is also depended on:

1. Place and type of implementation of IDMCS for that engineering design area
2. Definition of additional knowledge engineering issues based on technological features.
3. Definition of user/designer interface based on user profile.

Section 7.3 describes the methods of mismatch control for aerospace engineering design in details.

The stages of IDMCS development are illustrated in Fig. 17 and the features of detail stage of IDMCS development outlined in Fig. 18.

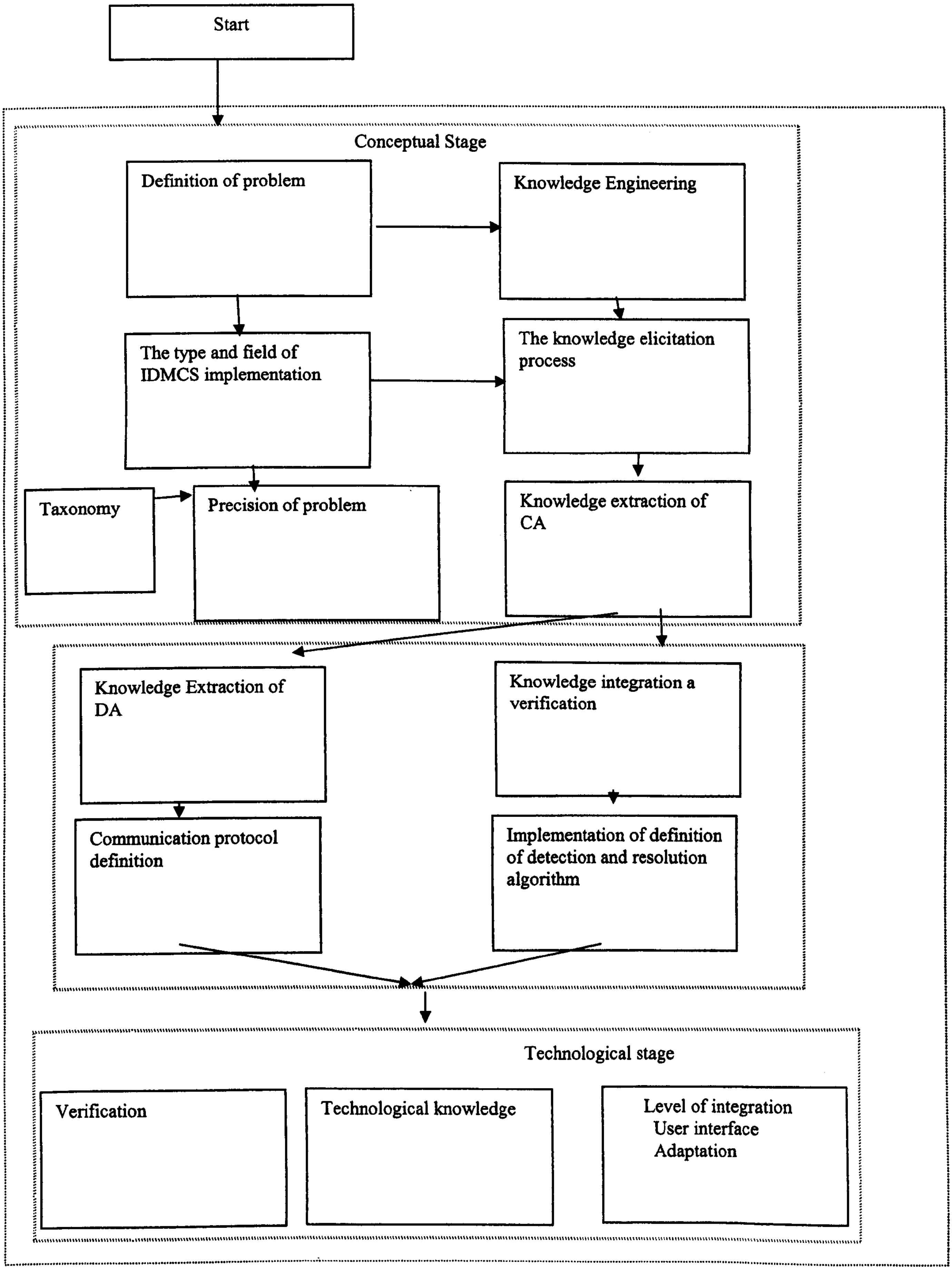

Start
Conceptual Stage
Definition of problem
Knowledge Engineering
The type and field of IDMCS implementation
The knowledge elicitation process
Taxonomy
Precision of problem
Knowledge extraction of CA
Knowledge Extraction of DA
Knowledge integration a verification
Communication protocol definition
Implementation of definition of detection and resolution algorithm
Technological stage
Verification
Technological knowledge
Level of integration
User interface
Adaptation

Figure 17: The IDMCS development scheme

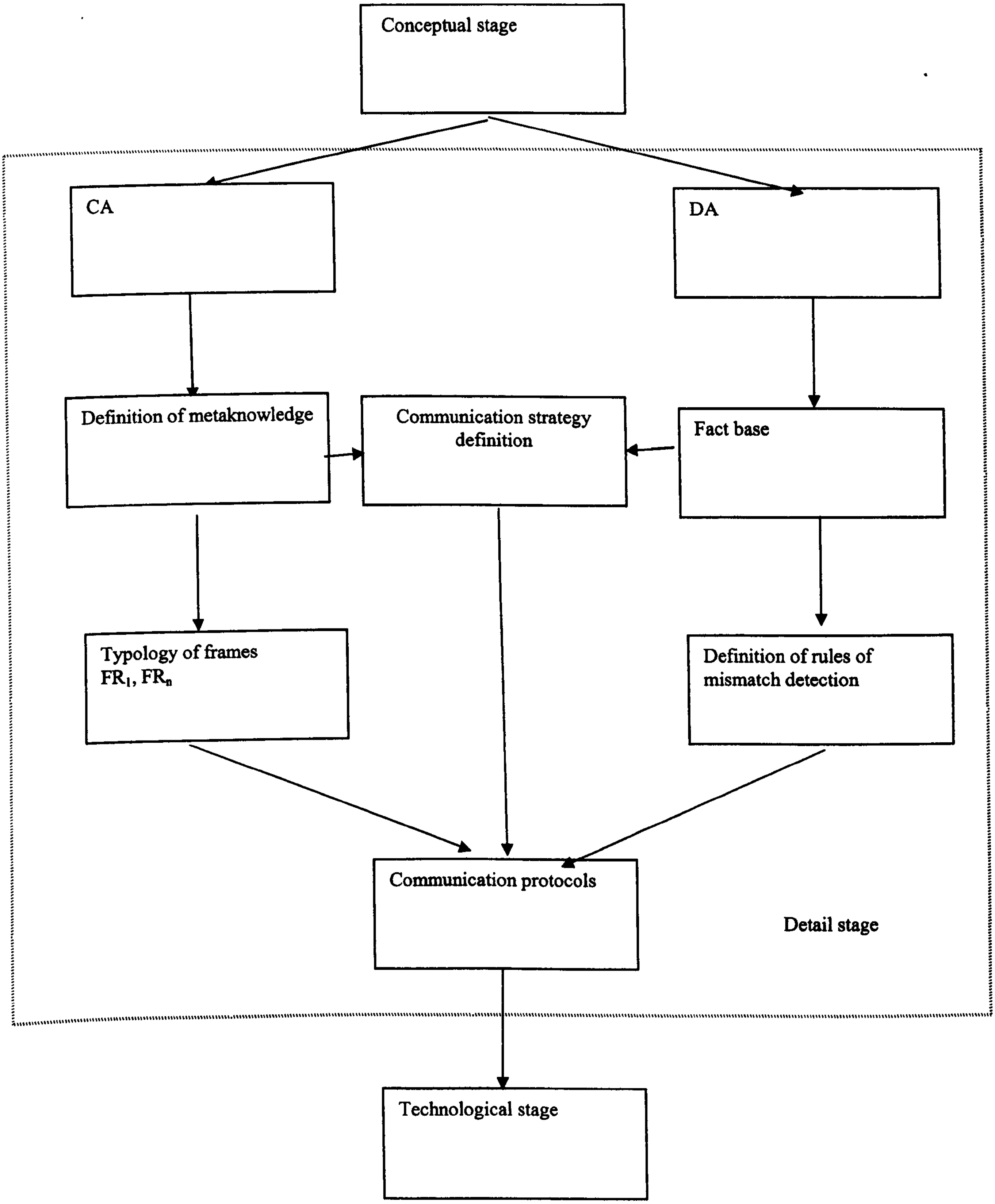

Conceptual stage
CA
DA
Definition of metaknowledge
Communication strategy definition
Fact base
Typology of frames FR_1, FR_n
Definition of rules of mismatch detection
Communication protocols
Detail stage
Technological stage

Figure 18: The relations between elements of detail stage IDMCS development

The elements of IDMCS development are described below.

Initial (Conceptual Stage):

- **Definition of problem**

At this stage the definition of requirements for IDMCS development is developed. We need to as certain whether it is really necessary to use an advanced IDMC capability to deal with wide number of design mismatches or we need only basic tool, for instance for tolerance analysis or assembly schedule definition. The number of tools can provide these capabilities especially for small and centralised design projects. In this case the development and implementation of IDMCS may not be absolutely necessary. But, if we have a collaborative design which involves a distribution of designers and managers, complex product – for example for aerospace or automotive sectors, the different number of manufacture and assembly mismatches, additional requirements such as cost, stress, corrosion, etc. the development and implementation can give a real improvement to the design process.

- **Knowledge Engineering**

Knowledge engineering aspects of IDMCS development based on knowledge elicitation principles.

The elicitation methods used to obtain the information from domain expert. In case of IDMCS the domain experts are first of all the designers from Design Departments and Managers of Design Centers. For the understanding of processes and Industry requirements for meta – level, the knowledge of Chief Designers is needed. There are many knowledge elicitation methods, such as case study, interview simulation, role-playing, prototyping, critiquing etc.

In industrial environments, combination of the interview and observation can be an appropriate solution for knowledge base development. Interview consists of asking the domain expert questions about mismatches related processes and how they perform the tasks.

Interview can be structured, unstructured and semi-structured [Foddy, 1995]. In our case the semi-structured interview is an appropriate solution and will be described in details in section 6.3.

- **Type and level of IDMCS implementation**

The definition of level of implementation of IDMCS based on classification of IDMCS by the level of engineering designer support during design process described in section 4.2 and defined as Interpreters, Advisory Systems and Prescriptive Systems.

The organisation knowledge-base for different classes of IDMCS can be very different and related to different levels of knowledge abstraction and interpretation.

Detail Stage

At this stage Fig. 18, a software agent's organisation is defined as well as the communication language. The DA and CA agent's communication algorithms are developed as well as knowledge base structure and fact bases. For IDMCS the typology of frames is also defined.

Metaknowledge base and Knowledge base

The development of meta and knowledge bases at the detail stage is about structural analysis of knowledge extracted from domain experts. Knowledge will be transformed to rules and frames using a knowledge engineering tool and process modelling software such as IDEF, UXL and etc.

Information exchange and communication protocol

Using the process map the possible relations between software and human agents should be investigated and the appropriate strategy of communication for different layers will be identified. If the strategy is not part of standard communication methods the number of agent building tools can provide the facilities to develop your own communication strategy.

The **technological stage** of IDMCS development for aerospace industry will be described in details in Chapter 7.

In this section the principles and methods of IDMCS development are introduced. We will use these solutions for definition of IDMCS architecture and organisation in next sub-chapter.

6.2 The IDMCS architecture and organisation

6.2.1 The general architecture of IDMCS

The IDMCS application focus is on complex large products in the field of aerospace or automotive industry. It is the distributed knowledge-based design support system, which uses the IDMC-approach, described earlier.

When designing complex large systems, the following steps are performed: (1) analysis of assembly parts (2) analytical evaluation of assembly possibility-Collisions and Minimum Distance Analysis, (3) choosing the script (conditions) of virtual –mock-up, and (4) progress analysis and generation of results.

The system analyses designer requirements to the design project given in the form of geometric 3D information and processes at the level of the distributed knowledge base.

A prototype IDMCS is developed using the ZEUS toolkit. The IDMCS overview is shown below.

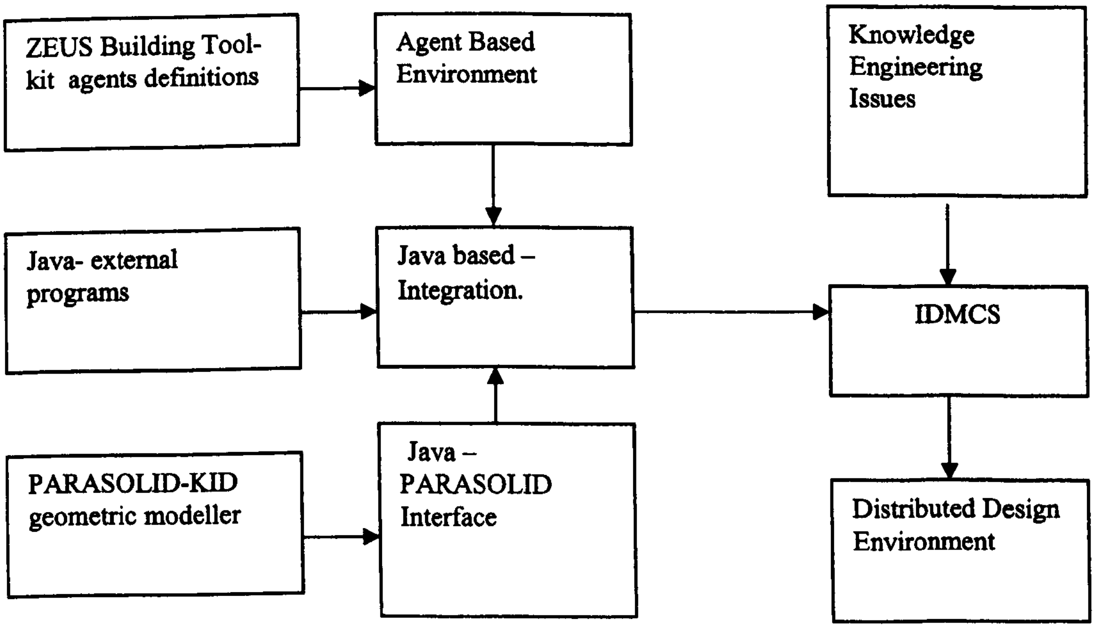

ZEUS Building Tool-kit agents definitions
Agent Based Environment
Knowledge Engineering Issues
Java- external programs
Java based – Integration.
IDMCS
PARASOLID-KID geometric modeller
Java – PARASOLID Interface
Distributed Design Environment

Figure 19: Development of IDMCS

As can be seen from Fig. 19 for the development of IDMCS we used JAVA programming language, ZEUS – Java-based toolkit from BTexact, and Parasolid - geometric kernel.

The user interface of the prototype system provides access to design and Control agents, visualisation of agent-based framework using the service functions of ZEUS Visualiser tool. The general structure of the user interface is outlined in Figure 20. An example of using the Agent Editor and Visualiser for aerospace design is outlined in Chapter 8.

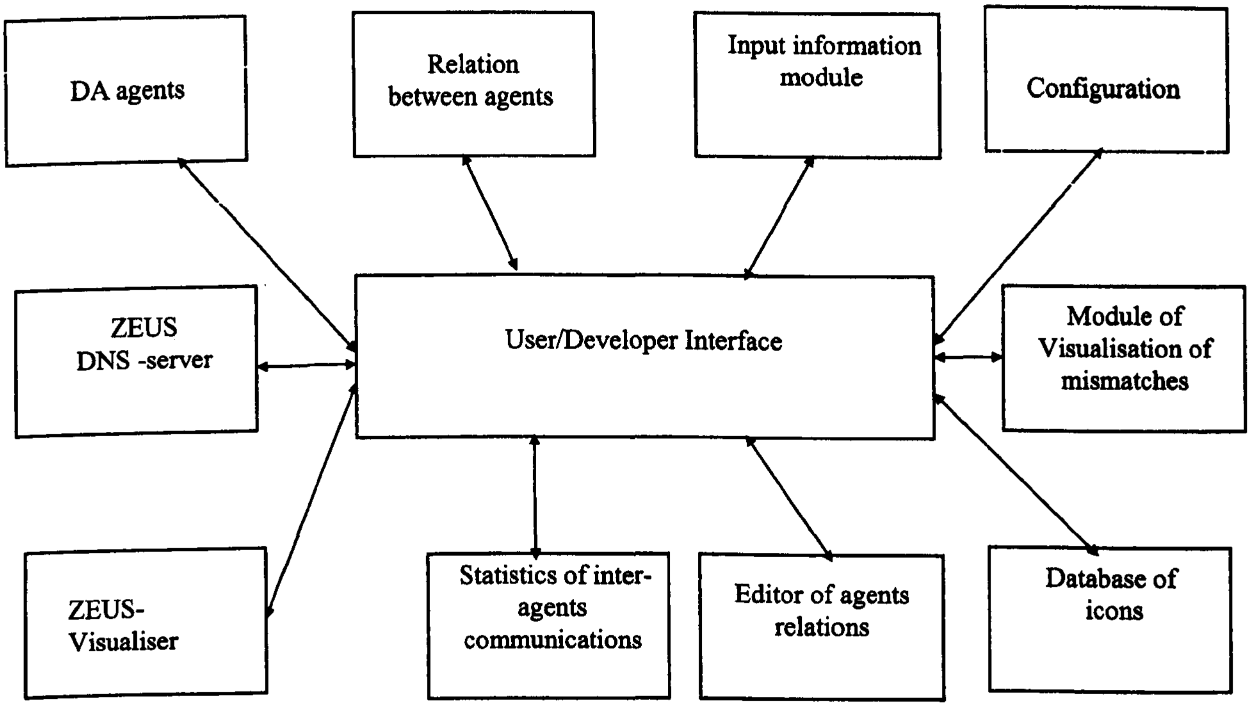


Figure 20: The elements of IDMCS User and Developer Interface

Integration of ZEUS and Parasolid

Integration of ZEUS and Parasolid was developed using Java sub-program which translates the geometric and material information L_{ig} from Parasolid to ZEUS IDMCS. After mismatch detection, the information about mismatch location - L_{loc} (geometric location, number of sub-assembly, material number) from ZEUS sends to Parasolid kernel (T-translator module). Figure 21 shows the general structure of the interface between ZEUS and Parasolid.

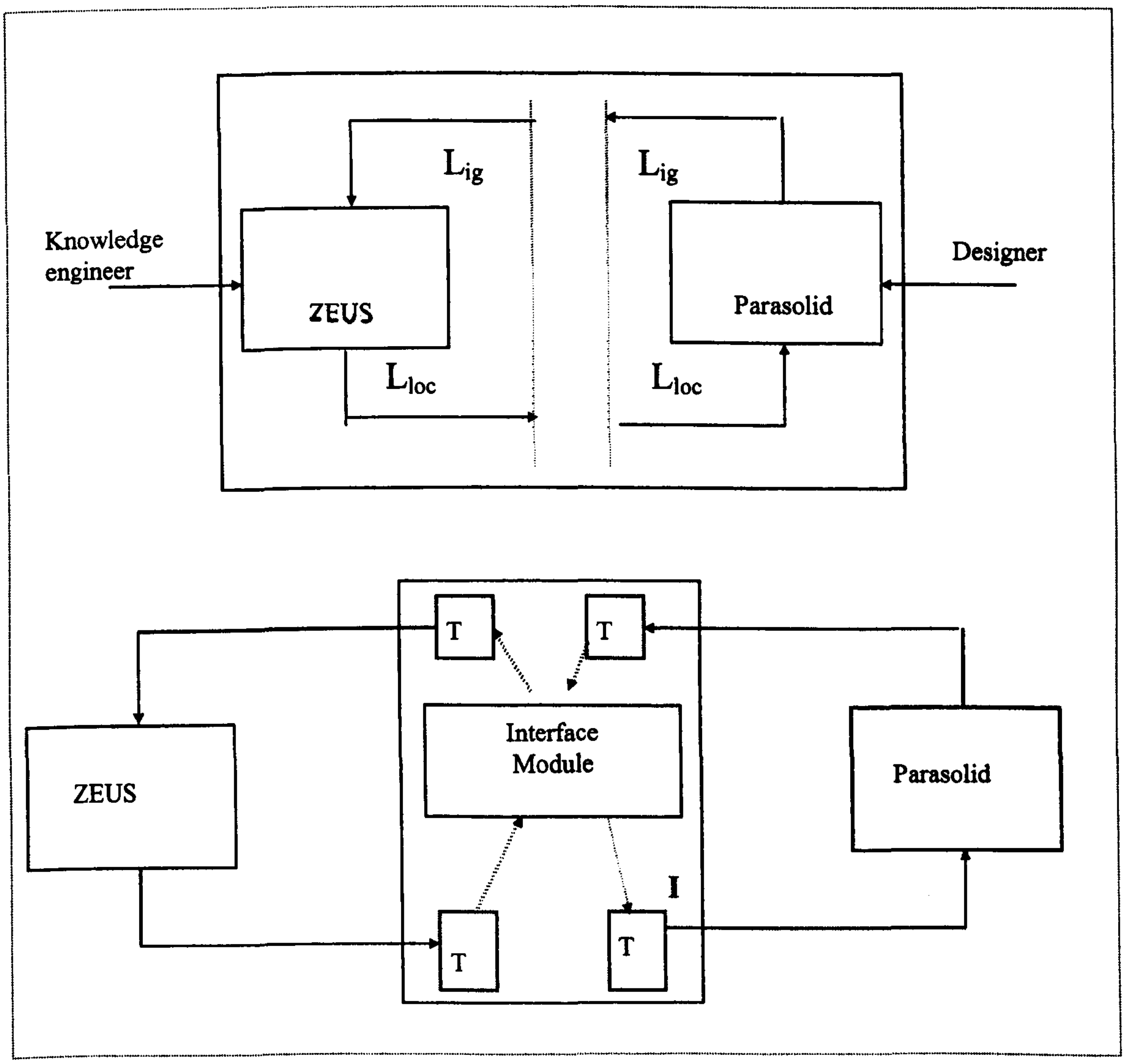


Figure 21: The Principles of Integration ZEUS and Parasolid

6.2.2 The protocols of negotiation

The possible communication framework between designers, agents and IDMCS was represented in Chapter 5.

This part of thesis reviews the possible scenario of negotiation with attention for possible organisation of Human-Computer Interaction in IDMCS.

In IDMCS we can define three types of users: engineering designers, managers of project, knowledge engineers/experts - Fig. 22.

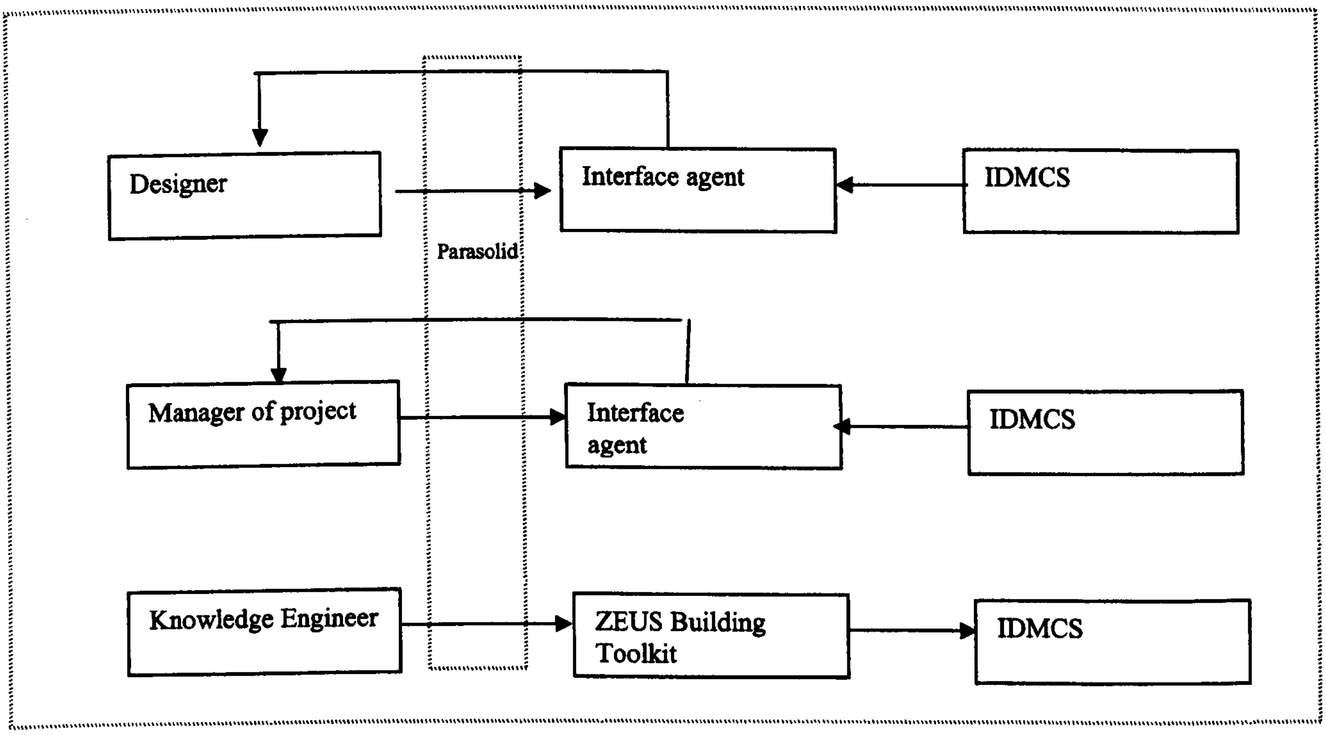


Figure 22: User Profiling using an adaptive Interface

The engineering designer/manager work with IDMCS, which can be physically distributed across different workplaces, buildings, and countries. Because of the physical distribution of the system, the negotiation with the user is the responsibility of **interface agent**. The Interface agent analyses the requests of information from designers or managers, passes on the requests to appropriate agents, organises the filtration of information according to the profile of designer/manager.

Other words, the interface agent is responsible for modification of output and input information according the to user profile. Interface agent is not part of ZEUS toolkit and can be developed as external JAVA based program.

The different situation of updated of IDMCS distributed knowledge-base using knowledge engineer (Fig. 22). This process is going off-line and knowledge engineering deals with ZEUS Agent Generator without using the interface agent.

The next section reviews the process of developing distributed knowledge-base for IDMCS and analyse the process of elicitation of distributed knowledge.

6.3 The distributed knowledge - base (Language L4)

The Structure of knowledge for distributed mismatch control

6.3.1 The Structure

In this part the structure of knowledge for distributed mismatch control (See Fig. 23), based on structure of design process from Table 7 (Section 5.3) is introduced.

As you can see from Figure 23, the number of issues should be considered at the stage of knowledge engineering process, such as, historical data about design mismatches handling in organisation; methods and tools of detection and handling of design mismatches; typical design/redesign requirements; suppliers mismatches; relations between cost and mismatches [Roy, R., Baker V., Griggs T., 2002], etc. On the other hand the knowledge engineering process is a process of capture of additional industry requirements for IDMCS and verification of theoretical framework.

We can define the knowledge needed for IDMCS as:

$$K(IDMCS)= < K_{his}, K_{design}, K_{proc}, K_{sup}, K_{ad} >$$

were:

K_{his} -historical knowledge;

K_{design} – knowledge about design requirements and design process;

K_{proc} -knowledge about the mismatch handling process;

K_{sup}, - knowledge about mismatches from suppliers; and finally

K_{ad} - additional knowledge

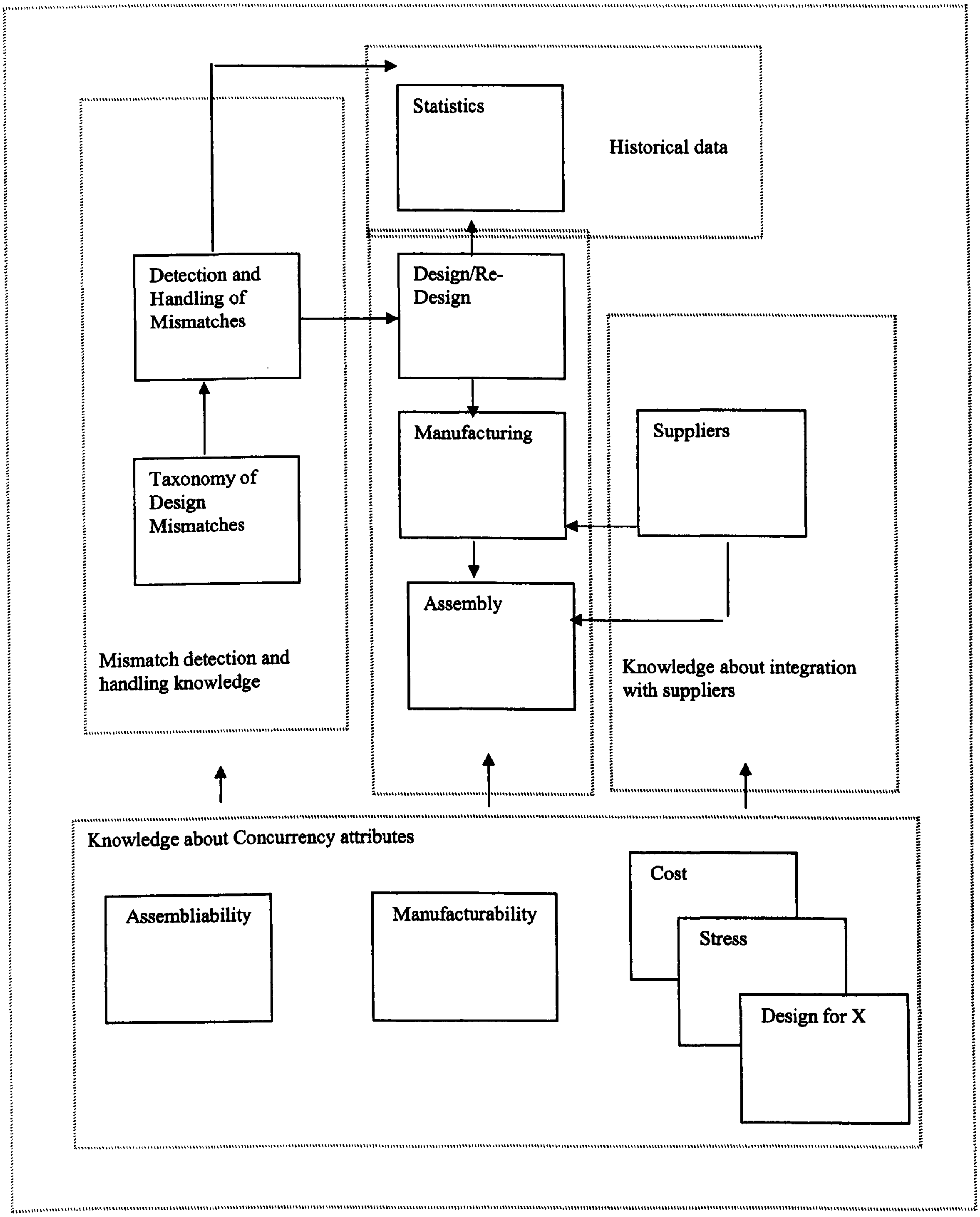

Statistics
Historical data
Detection and Handling of Mismatches
Design/Re-Design
Manufacturing
Suppliers
Taxonomy of Design Mismatches
Assembly
Mismatch detection and handling knowledge
Knowledge about integration with suppliers
Knowledge about Concurrency attributes
Assembliability
Manufacturability
Cost
Stress
Design for X

Figure 23: The structure of knowledge about mismatch control process

As briefly described in section 6.1 the interview method used for knowledge elicitation, is based on semi-structured interview techniques [Foddy, 1995].

Unlike the detailed questionnaire, where detailed questions are formulating ahead of time, semi-structured interviewing starts with more general questions or topics.

Relevant topics (such as mismatch detection methods, statistics, mismatch and suppliers) are initially identified and the possible relationship between these topics and the issues such as mismatch historical data, relation within design Department, etc, become the basis for more specific questions which do not need to be prepared in advance.

In this research the combination of observation and questioning was used for the development of knowledge-base of IDMCS.

The next section represents aspects related to the organisation of the distributed knowledge-base.

6.3.2 Design mismatches identification. Knowledge extraction and Interview methods

I would like to acknowledge the support of my second supervisor Dr Martin Stacey in developing of this questionnaire. His research experience in psychology of design process was extremely useful.

The semi-structured interview method was used to identify the rules for IDMCS base, for design and control agents. A structure of questionnaire is represented below and the full questionnaire is outlined in Appendix B.

The questionnaire includes ten parts such as:

- Statistics
- Mismatches and Suppliers
- Detection of Mismatches
- Handling of Mismatches
- Negotiation
- Decision making
- Re-design
- Mismatch detection/handling Software
- Distributed design process
- Interview

The structure of semi-structured questionnaire is presented in Fig. 24.

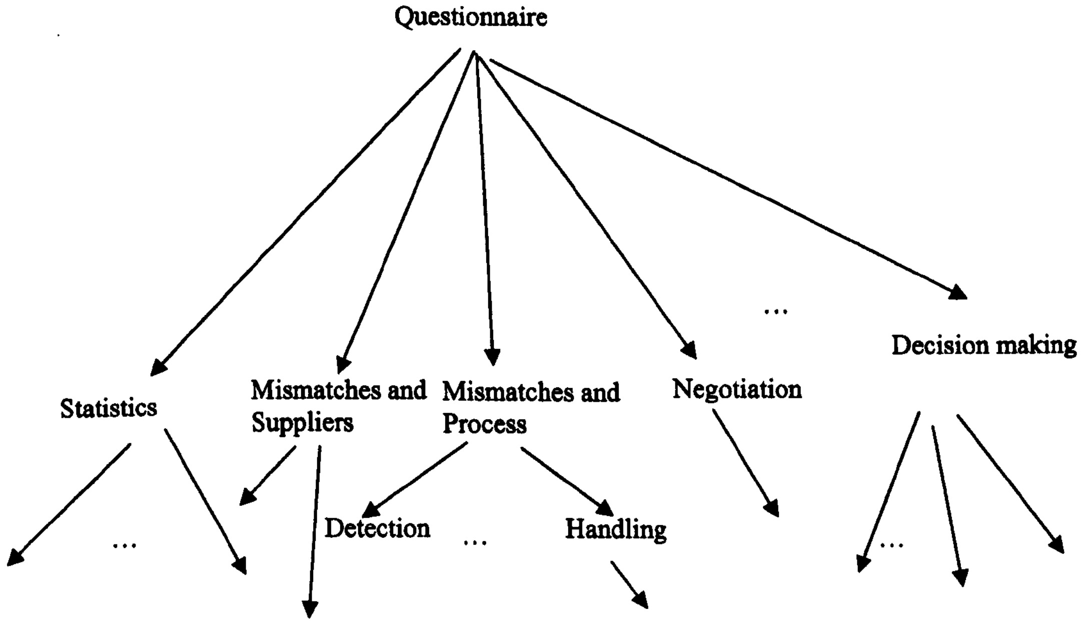


Figure 24: The structure of questionnaire

The questionnaire is based on a number of questions described below:

Table 7: An example of semi-structured questionnaire for IDMCS.

No. of question	**Part of questionnaire...**	**Question**	**Possible answers**
1	**Statistics**	When does Mismatch typically occur?	Geometrical Material Manufacturing Assembly Avionics ?
2	**Statistics**	Can you give a rough estimate of the percentage of cases falling into each category?	?
3	**Statistics**	What is the cost/resources associated with it?	Hours ? Money ?
4	**Statistics**	Can you give me an example,	?

No. of question	Part of questionnaire...	Question	Possible answers
		ideally an example of a mismatch in each group?	
5	**Mismatches and Suppliers**	What kind of suppliers: - in which industry sector? - between which industry sectors? - between which countries?	?
6	**Mismatches and Suppliers**	Have you got any idea why?	?
7	**Mismatches and Suppliers**	Who defines the interfaces between components?	?
8	**Mismatches and Suppliers**	What measures are taken to avoid mismatches? - contrast to in-house procedures?	?
9	**Detection of Mismatches**	Who typically detects mismatches?	-Computer? -Manager? -Designer of one component? - Designer who uses a mismatched components ?

No. of question	Part of questionnaire…	Question	Possible answers
10	**Detection of Mismatches**	When? - what is typical - when is the latest - when should they be detected	
11	**Handling mismatches**	What is the procedure in your company for dealing with mismatches?	
12		Is this the same across the company?	
13	**Negotiation**	Whom do you negotiate with?	- Managers? - Designers who designed part? - Designers who do rework?
14	**Negotiation**	Does the negotiation involve all concerned parties?	?
15	**Negotiation**	How formal are they? Who participates? What is considered (do you try to solve the problem together, or talk about what the	?

No. of question	Part of questionnaire...	Question	Possible answers
		problem is, or what)?	
16	**Negotiation**	What are the outcomes of the discussions? (Any decisions, or progress towards deciding what's possible, or being better informed about the nature of the problem?)	
17	**Negotiation**	If there are negotiations, who arbitrates negotiations?	
18	**Decision making**	Who decides what should be changed?	
19	**Decision Making**	Does he/she have technical competence?	
20	**Distributed design**	How many companies abroad part of your Consortium?	
21	**Distributed Design**	Language barriers?	
22	**Distributed Design**	What is the effect on the design process or on mismatch handling when organisation is distributed?	
23	**Background**	Interviewee age, experience,...	

The main respondents were people from industry, who currently involved in engineering design process.

Some information about respondents is summarised in Table 8.

Table 8: Interview process. The persons involved.

Management level	**Position**	**Number of persons**
Top	Constructor General	1
Top	Vice-president in Design Technology	1
Senior	The Head of Engineering Design Centre	1
Senior	The Head of Engineering Department	1
Senior	The Head of Technology, Main Technologist	1
Senior	The Head of CAD Centre	1
Engineering	Design Group Manager	2
Engineering	Designers	9

Based on knowledge elicitation process, the knowledge bases for D-and C-agents were defined and the taxonomy of mismatches was verified.

6.4 Conclusion

In this chapter the main principles of organisation of IDMCS were explained. The System's development and overview are presented.

In many areas software engineering methods have been developed. Multi-agent approach is a new paradigm of software organisation and new methods of design and such sort products are needed.

To develop the IDMCS we need the three main stages such as initial definition (conceptual stage), detail stage and technological stage.

At the initial definition stage the knowledge engineering issues are defined, types of mismatches needed to be resolve, using taxonomy of design mismatches.

Detail design stage: At this stage the negotiation process methods, type and dimension of multi-agent frameworks, agent negotiation algorithm, set of CA and DA agents, mismatch detection and resolution algorithm are defined.

The technological stage is more about how to implement the theoretical framework for real life problem and implementation of IDMCS for described Industry sector, field, e.g. Aerospace, automotive, communications, defense.

The technological stage is really depends from the field of IDMCS implementation and defined taxonomy of mismatches and very different in:

1. Place and type of implementation of IDMCS for that engineering design area
2. Definition of additional knowledge engineering issues based on technological features.

3. Definition of user/designer interface based on user profile.

The IDMCS architecture and organisation is outlined as well as the protocol of negotiation between IDMCS and users during mismatch control process. The Integration of ZEUS and Parasolid was described.

Also in this chapter the knowledge engineering issues were analysed including an example of semi-structured questionnaire used in knowledge elicitation process.

The principles of organisation described in this chapter were helpful for using IDMCS for design support in aerospace.

The next chapter is described the mismatch control in aerospace design.

Chapter 7 Aerospace Design and Mismatch Control

7.1 The Principles of Aerospace Design and Mismatch Control in Aeronautics

One of the key challenges for Europe is to maintain and develop the European Aerospace sector as a world competitive industry [The European Aerospace Industry..., 1997]. The European Commission (EC) has fostered several collaborative research initiatives in aeronautics yielding a number of successful projects [AIT Initiative; Design Process in Aerospace Industry; Multi-site Concurrent Engineering ...; Bechkoum, 1997; Smith, 1999]. In the Fifth Framework Programme of the EC the financial support dedicated to the Aerospace industry alone is set to 700 million Euro.

This increased financial support reflects the need for the aerospace industry to make use of emergent technologies that enable an integrated approach for European cooperation [Bond and Ricci, 1992; Bradford, 1995]. To this effect restructuring activities are underway and core clusters for activities are forming between partners in the sector. The need for a more coordinated cooperation is not a new phenomenon. For many years the design and manufacture of major European aerospace products has been distributed across the continent; Airbus and EFA being typical examples. What makes cooperation amongst partners of the European aerospace sector more challenging is the fact that the design process tends to be sequential and requires centralised planning teams and a great deal of travel on the part of the distributed designers. The situation where multi-disciplinary expert teams have to travel too frequently from one organisation to another and stay away from their working environment for long periods presents deficiencies in both costs and quality terms.

Research in collaborative design is described by [Favela et al., 1993; Ganesham and Prakash, 1996; Gascoigne, 1995; Hardwick et al., 1996; Kock, 2000; Lu and Udwadia, 1999]. Examples of typical problem areas include:

- difficulty in planning for and organising meetings
- loss of expensive man-hours in meeting preparation and journey
- unavailability of tools familiar to an invited expert group at a given host site
- frequent lack of productivity, for example because of non-homogeneous design levels between the participants or because document items, necessary for discussion, have been forgotten (e.g. "*Sorry, I was not aware that I needed to bring this document with me!*").

CSCW techniques promise to resolve most of the difficulties above by replacing the *paper and tape* and *physical meetings* based methods by electronic communication and electronic meetings.

This contrasts sharply with procedures in other regions, particularly the USA, where the design is often kept in one main location even when the components are manufactured elsewhere and transported to a main assembly plant. The centralisation allows for the relatively easy introduction of concurrent engineer design practices that reduce design cycle time. But now the problem of distribution of design and manufacturing processes is becoming important for American aerospace as well. This is because the global recession and impact of September 11th attack ruled the USA giants such as Boeing to try to find new solutions to reduce the cost of design and manufacture for new products using foreign suppliers and designers.

The next Figure (Fig. 25) shows the international distribution of design and orders for Boeing and AIRBUS.

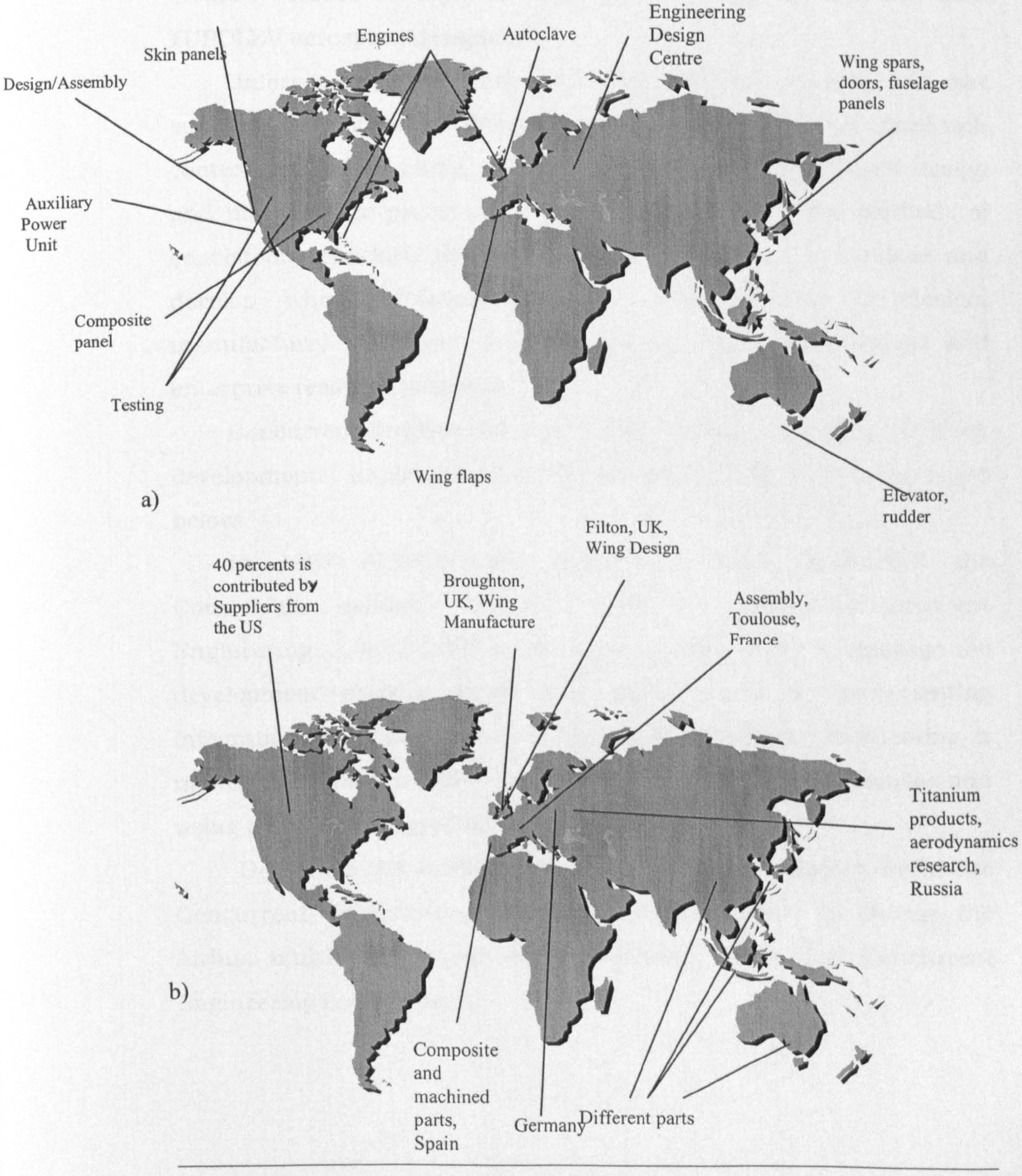

Engineering
Design
Centre
Engines
Autoclave
Skin panels
Wing spars, doors, fuselage panels
Design/Assembly
Auxiliary Power Unit
Composite panel
Testing
Wing flaps
a)
Elevator, rudder
Filton, UK, Wing Design
40 percents is contributed by Suppliers from the US
Broughton, UK, Wing Manufacture
Assembly, Toulouse, France
Titanium products, aerodynamics research, Russia
b)
Composite and machined parts, Spain
Germany
Different parts

Figure 25: Boeing (a) and AIRBUS (b) design and manufacture across the Globe

Boeing already has established the Engineering Centre in Moscow, Russia using the high profile skills of ILUSHIN and TUPOLEV aerospace designers.

Unfortunately the current CAD/CAM/CAE systems do not support the distributed design process in full as well as mismatch control process as noted earlier. To organise the distributed design and manufacture process is important to have not only methods of consistency checking discussed in this thesis, but to analyse and develop whole extended enterprise infrastructure for design, manufacture, mock-up, re-design, product data management and enterprise resource planning.

Concurrent Engineering approaches can be a milestone for those developments. Important research projects in this area are outlined below.

In 1997 AIRBUS and Aerospatiale have established the Consortium research project MUSCLES [Multi-siteConcurrent Engineering...]. MUSCLES is to provide a methodology to redesign the development process within the Airbus partners, implementing integration of human resource management, process engineering & management issues and ICT enablers according to CE principles and using distributed Digital Mock-Up techniques.

Delivering the basic skeleton and tools for complex multi-site Concurrent Engineering, MUSCLES intends mainly to change the Airbus multi-site present way of working into a full Concurrent Engineering environment.

Practically, MUSCLES will deliver a full set of tools and methods to redesign the development process of a complex product in a multi-cultural and multi-site environment.

Earlier I mentioned project INDEMAND from Cranfield University and British Aerospace - Integrating Design and Manufacturing Knowledge in an Extended Enterprise. INDEMAND is oriented towards Current manufacturing practice to increase the outsourcing of component manufacture to external suppliers. The roles and relationships in the supply chain are progressively changing. In many engineered products around 70% of the value is contributed by external suppliers. In some engineering industries, such as the UK Aerospace Industry, extended enterprises exist in which the Product Owner has predominantly become a designer and assembler.

To achieve effective design for manufacture in an extended enterprise, the design team needs to know the limitations of the manufacturing capabilities of suppliers in the potential supply base for a component.

As can be seen from the outlined projects, the Concurrent Engineering methodology for aerospace production has significant differences from automotive industry and mechanical engineering in general. This is because the number of items (cars, planes) for automotive industry can be 100000, but for aerospace only hundreds, but complexity of aerospace products is much higher and requirements are very high as well.

The next figure from [Niu, 1999] presents the Aircraft design and manufacture process. We will modify the picture to show the place of IDMCS in aerospace design and manufacture process.

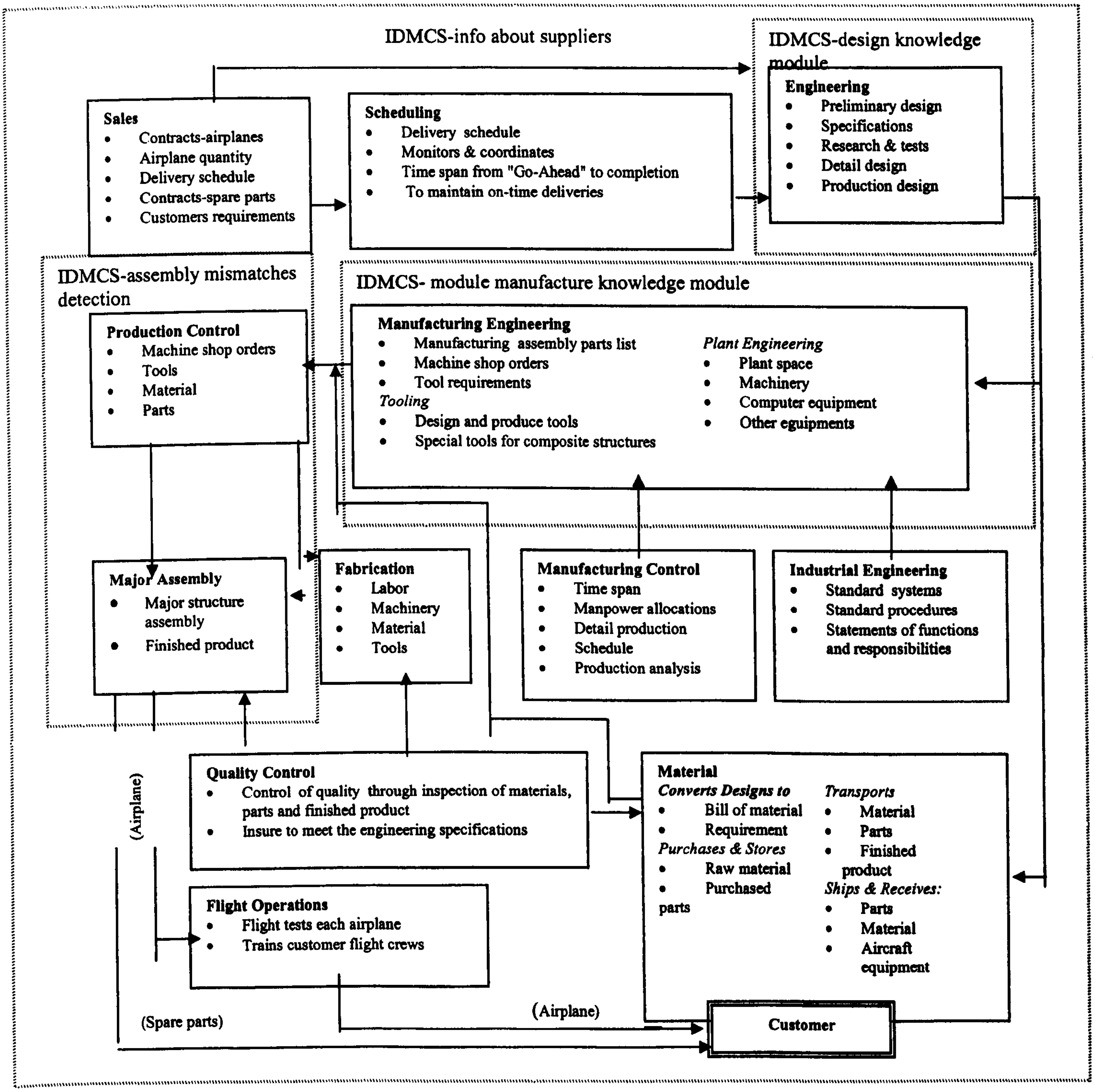


Figure 26: The aircraft design and manufacture process

As you can see the mismatches in aerospace manufacture and design process allocated not only in main production circle, but also depend on suppliers, stages of quality control, the testing of the system.

The level of inter-elements communication in design – production circle is very significant. At the present time the IDMCS functionality related to design, manufacture and assembly control process.

For instance, we are using IDMCS and DFA/DFM taxonomy for development for a distributed knowledge-based design support system which detects geometric and material irregularities at the assembly stage of aerospace design.

The IDMCS provides mismatch control during wing-box assembly process, using an initial set of data from aircraft design sources [Nui , 1999; Raymer, 1999; All sets for more wings ...; Torenbeek,1982; Automated wing box assembly..., 1999; Butler, 1998; Daberkow and Marvis, 1998; Ford, 1998; Hill, 1997; Kolb, 1994; Mohammad et al, 1996; Quayle, 1999; Reithmaier, 1991; Voit et al., 1987] and AIRBUS and Electroimpact Inc. design engineers.

When designing using IDMCS, the following steps are being performed: (1) analysis of assembly parts - assembly checks of stringers, skins, spars etc., (2) evaluation of assembly possibility - Collision and Tolerance Analysis, (3) manufacturability analysis, (4) choosing the alternatives for mismatch resolution, and (5) semiautomatic mismatch resolution and generation of results. .

It is possible to extend the role of IDMCS for stages which are part of production planning methods, Enterprise Resource Planning or testing processes such as: Manufacturing Control (mismatches of schedule, manpower allocation mismatches, etc), Virtual testing-simulation (before real flights experiments) - mismatches of work, Shipping and handling processes with suppliers.

Some of the possible future developments of IDMCS will be presented in the Conclusion part of the thesis.

7.2 The specific knowledge about design process. Aerospace design example

In section 6.1 the stages, of IDMCS were described. The technological stage, as case specific is described in this part of dissertation. The technological part of IDMCS includes:

- The definition of general field of implementation. Industry sector – for instance aerospace, automotive, electronics, etc
- The definition of areas within industry sector, such as satellite design, aircraft design, helicopter design, etc.
- The definition of areas within specific design field for elicitation of technological knowledge, such as wing box design, fuselage, landing gear design, and etc.

The case specific knowledge is part of D and C agent's knowledge-base.

The example of aerospace related processes and heuristics would be used for extension of basic taxonomy of design mismatches and development of taxonomy for aerospace design.

The examples of such heuristics annotated in [Niu, 1999] and are shown in Table 9.

Using these heuristics we can define the field of knowledge in knowledge-base of IDMCS related to each of these rules.

This solution is summarised also in Table 9.

Table 9: The case specific knowledge for IDMC. Aerospace design example

No.	**Rule/Recommendation [Niu,1999]**	**Part of IDMCS Knowledge base**	**Knowledge needed**
1	Maxing of fastener materials and types in any one fastener pattern or area should be avoided	D-agent Rule-base	Materials Geometry
2	Tolerances less than ±003 for length, depth and width, and ±001 for machine thickness should be coordinated with the manufacturing.	D-agent Rule-base C-agent –manufacture process knowledge	Geometry
3	Make ribs normal to the front or rear spars where practical to minimize tooling and master tooling template problems.	D-agent Wing-box design	Geometry
4	Crawl holes through ribs and spars should be a minimum of 12 inches by 18 inches Larger holes should be used	D-agent	Geometry Diameter Stress data

No.	Rule/Recommendation [Niu,1999]	Part of IDMCS Knowledge base	Knowledge needed
	where allowed by shear stress in minimum gage areas. Consideration should be given to hole locations in adjacent ribs for maintenance. Sharp corners and protrusions around crawl holes should be eliminated.		
5	Aluminum alloy upset head rivets or pull-type lockbolts should be used for web and stiffener riveting wherever possible. Design should consider automatic riveting.	C-agent-manufacture process knowledge	Material Process knowledge
6	All designs should consider supplier capability, particularly in sizes and kinds of raw materials or standards, so that at least two sources are available. Competition for orders is thus maintained, and not	C-agent- supplier process information	Supplier detail size, raw material type, standards

No.	Rule/Recommendation [Niu,1999]	Part of IDMCS Knowledge base	Knowledge needed
	dependent on a single source in case of emergency.		
7	Manufacturing recommends stringer tolerances as follows Thickness ±001, width and length ±0 03, height ±0 03. Special deviations may be made on basic gage taper dimensions and cutter radii. All tolerances should be reviewed for weight savings within the established economic limits.	D-agent	Geometry
8	All skin tolerances should be as follows Thickness ±0005, edge mm and critical location coordinates ±0 03	D-agent C-agent	Geometry
9	Edge margins of rib cap to panel stringer attach bolts shall be standardized for each diameter of fastener. A standard tool can then be	C-agent-manufacture process knowledge Wing-box design process knowledge	Process knowledge Assembly schedule

No.	Rule/Recommendation [Niu,1999]	Part of IDMCS Knowledge base	Knowledge needed
	used for drilling these holes in the wing assembly fixture.		
10	Web stiffeners on ribs should be located to allow use of either bolts and nuts or pull-type lockbolts for rib cap to panel stringer attachment as far as is practical. This should be kept in mind for all areas. Make room for lockbolts and Hi-Look fastener equipment if possible.	D-agent	Detail orientation - geometry

Using the set of recommendations described in this sub-chapter we can define a case specific knowledge-based of IDMCS so we can combine the extended taxonomy of design mismatches of detection of mismatches and a sort of requirements for generation of the recommendations for designer.

Also the case specific elements of IDMCS development are related to levels of integration IDMCS and current Design tools within organisation, development of case specific user/designer interface, adaptation of distributed knowledge base strategies.

The next sub-chapter will review the development of extended taxonomy of design mismatches particular to aerospace design.

7.3 The Taxonomy of mismatches of aerospace design

Using the general framework for development of taxonomy of design mismatches described in section 4.3 we can define the case specific taxonomy. In this dissertation the taxonomy of mismatches in aerospace design is developed.

For better understanding and to reduce the complexity the proposed taxonomy is restricted by assembly and manufacturability mismatches. To define taxonomy, criteria of classification should be considered. In our case we have a main criterion - *assembly process*, and additional criteria as *types of connections* and *indicators* (critical parameters- M_{par}^{cr}) (described earlier). In this case taxons are assembly mismatches.

The bolted connection requires critical parameters M_{par}^{cr} (such as thread major diameter, minor diameter and pitch) to be in accordance.

For the correct mismatch detection process we need to represent into our knowledge-base geometric information and information about materials from which parts are prepared. The taxonomy is shown on next Figure 27, 28 and 29.

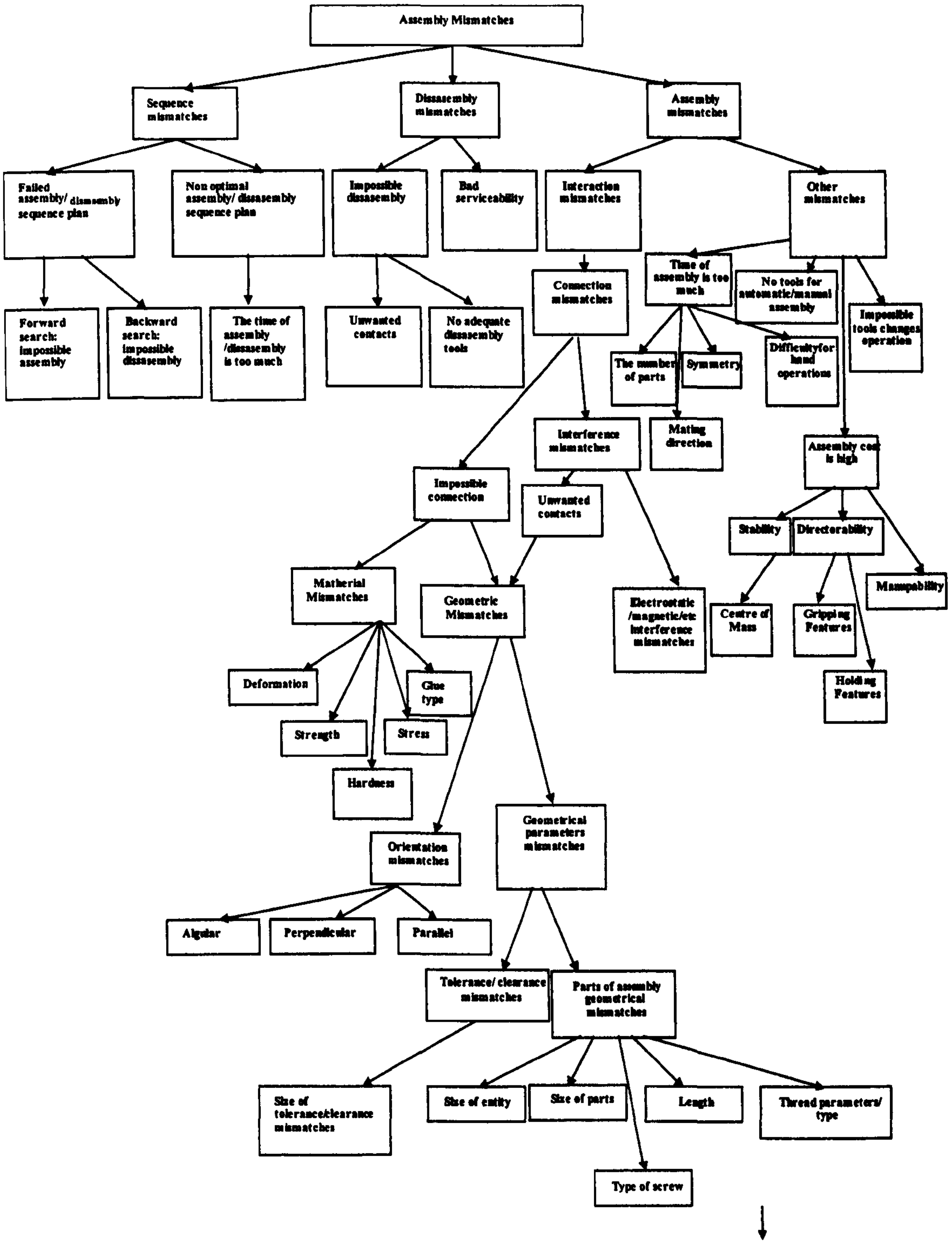


Figure 27: A Taxonomy of Design Mismatches

Fig 28, 29

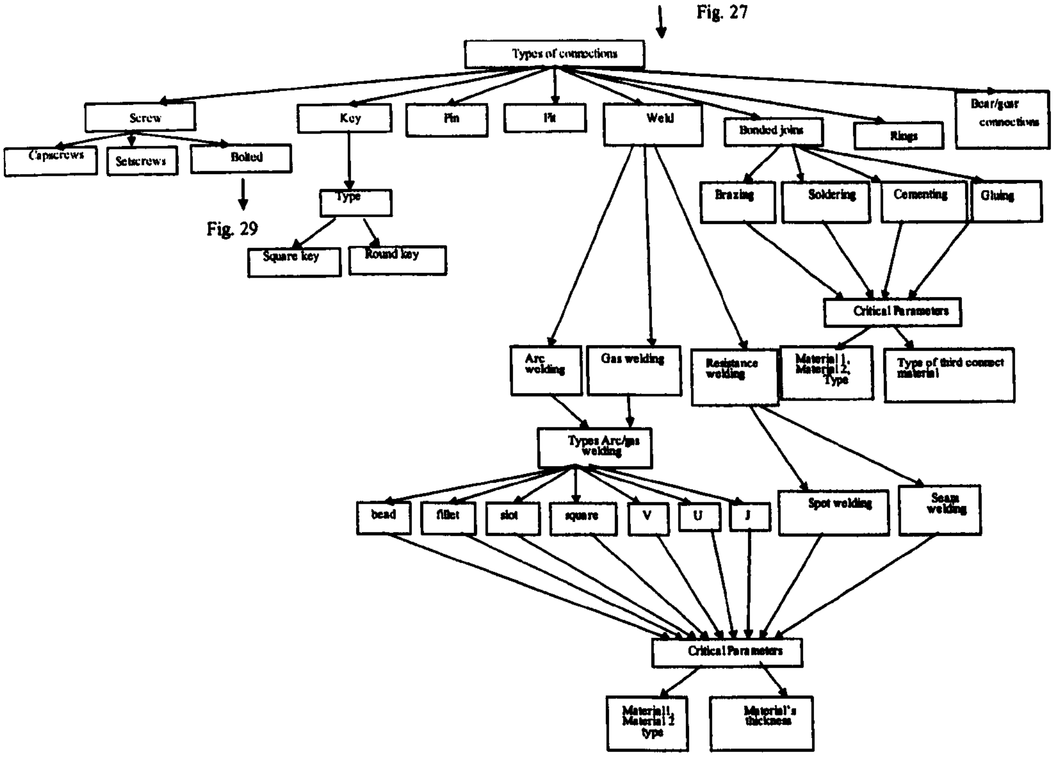


Figure 28: A Taxonomy of Design Mismatches. Types of Connections (cont.)

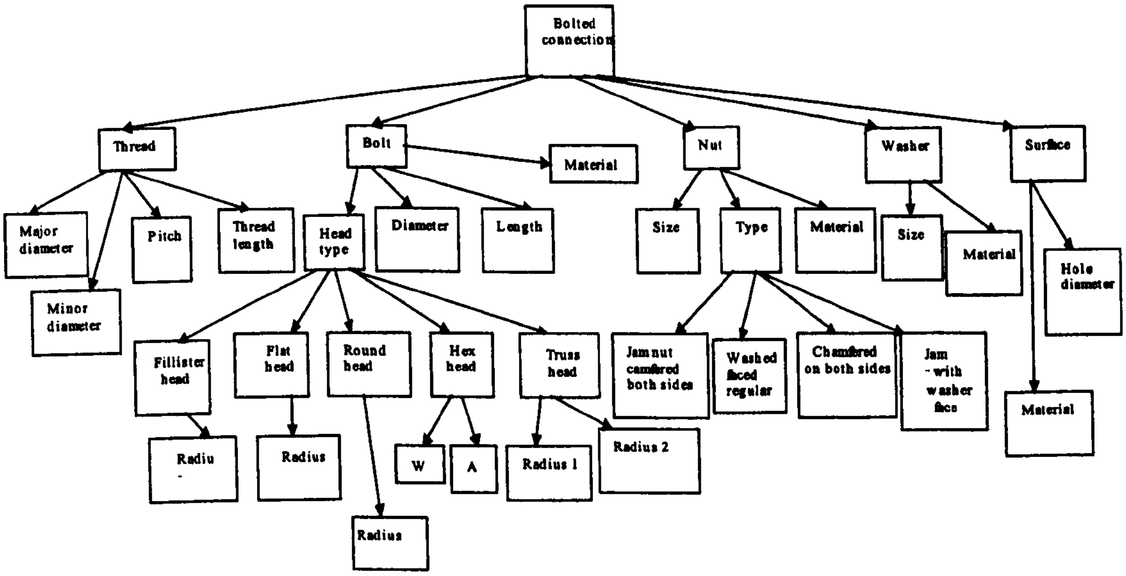


Figure 29: A Taxonomy of Design Mismatches. Bolted Connection.

This taxonomy was developed and verified through intensive interview process within Industry sector. This taxonomy is used for IDMC Industrial Case Study, for instance for mismatch control during wing-box design process.

7.4 Conclusion

This part described the use of Intelligent Distributed Mismatch Control for Aerospace Design. The principles of Aerospace design and Mismatch control are introduced.

The main principle is that distribution of design and product development is needed to develop the new methods of organisation of extended enterprise IDMC-approach can be an important part of this research as well as integration with Concurrent Engineering Methodology.

A number of projects such as MUSCLES, INDEMAND have paid attention to Concurrent Engineering environment research and present ideas how to integrate IDMCS and Extended Enterprise.

The technological stage of IDMCS development is described in this part of dissertation.

The technological part of IDMCS includes:

- The definition of general field of implementation. Industry sector – for instance aerospace, automotive, electronics, etc
- The definition of areas within industry sector, such as satellite design, aircraft design, helicopter design, etc.
- The definition of areas with in specific design field for elicitation of technological knowledge

The implementation of IDMCS as aerospace design and manufacture process is outlined.

The knowledge needed for definition of case specific knowledge-base of C-and D-agent is developed, as well as, the taxonomy of design mismatches relating to aerospace design process.

The next section reviews the industrial case studies and implementation of the results described in this dissertation.

Chapter 8 Industrial Case study. Experiments and Implementation

This chapter describes the industrial case studies and experiments undertaken in the research described in this thesis. It was performed under collaboration with major aerospace and aerospace related companies such as AIRBUS UK, Electroimpact Corp., USA, TUPOLEV Corp, Russia, AVIASTAR, Russia, and Euro-Russian Aerospace Consortium.

8.1 Using IDMC-approach for aerospace design and manufacture - wing-box design

8.1.1 Wing box structure, assembly, and manufacture

Following an implementation of the research carried out in the areas described above, a period of practical research ensued. With help from Electoimpact Inc. and British Aerospace Airbus, Broughton, during the visit Broughton BA facility general requirements for the assembly process were observed, using engineering and technological knowledge.

Practical issues related to work practice and to technology usage were considered.

At this stage a number of typical knowledge engineering procedures were organised, such as knowledge retrieval (using semi-structured interviews, texts analysis, and observation), structural analysis of knowledge (definition of terminology, terms and relations, attributes and etc.).

Underlying this research was the hypothesis that current design and technological systems do not support the requirements of collaborative concurrent engineering described earlier. The models were implemented for assembly mismatch control of wing box Airbus 340.

The major assembly begins with wing production. The wing design process is a very complex one [Ford, T., 1998; Bobrowski et al, 1999; Butler, 1998; Voit et.al, 1987; All set for more wings...,1999; Automated Wing Box assembly.., Butley, 1998]. Components of wing are installed and joined in a tool called a wing majors assembly (WMA). A wing box is the structural component of an aircraft wing [Knowledge Based Engineering at Airbus, 1998] (see Figure 30).

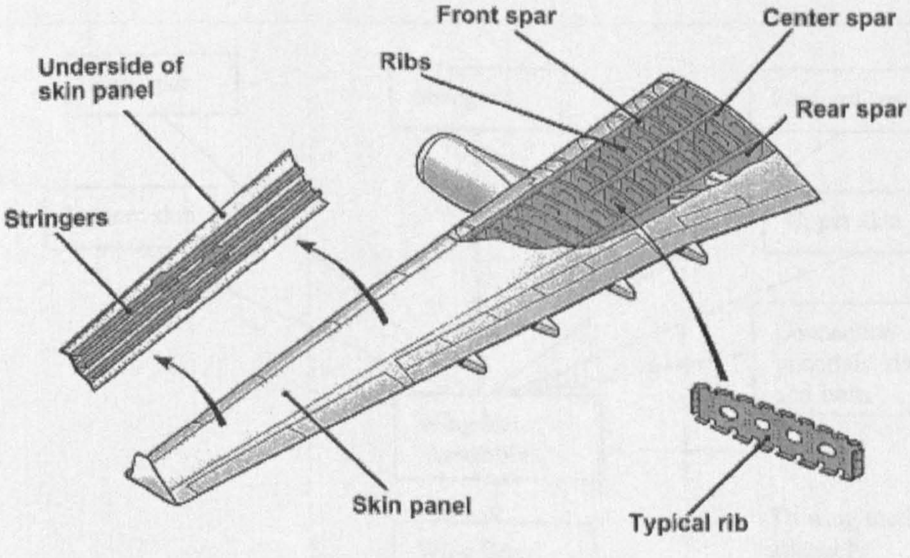


Figure 30: Aeroplane Wing Structure (© British Aerospace Airbus)

A wing section comprises wing box, panels, brackets, fasteners, fixed leading- and fixed trailing-edge sections.

The wing box model consists of several parts - stringers, skins, spars and ribs. In a large aircraft wing there can be over 50 ribs and 100 stringers. That means that there are a lot of rib feet in one wing. Geometric constraints and dimensions define the parameterisations and assembly relationships between the parts. For instance, spar components include upper and lower chords to support aluminum wing skins; a vertical web, a large, flat surface between the chords; and vertical stiffeners and rib posts extending across the length. The general assembly process is represented in Figure 31.

Figure 32 shows the assembled model [Knowledge Based Engineering, 1998]. The figure illustrates that the bolts in the wing pass through the rib feet, stringer and skin. It is important to provide tolerance analysis during final assembly.

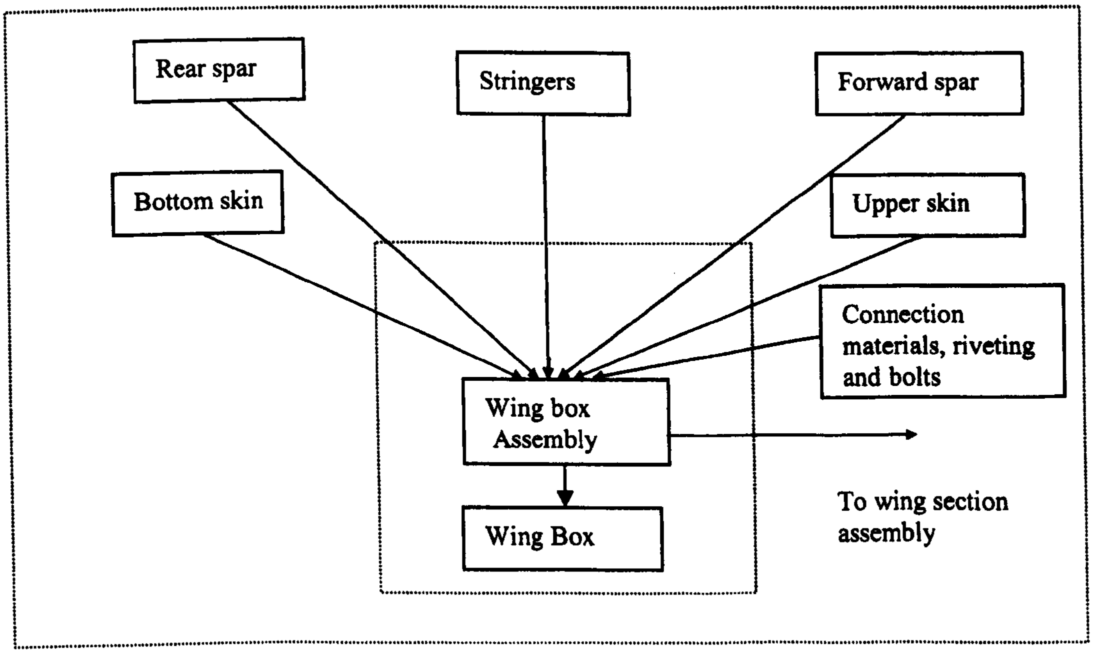


Figure 31: Wing Box General Assembly Process

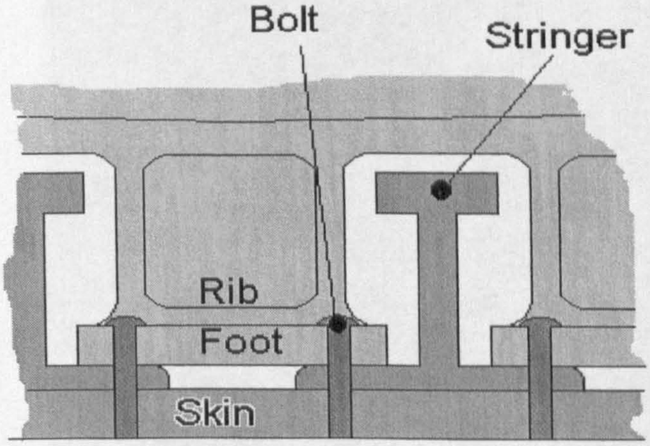


Figure 32: A340 wing's Connections (© British Aerospace Airbus)

The process of connecting the rib to the wing skin, using Electroimpact A340 Wing assembly process is automated, using a wide range of technology including numerically controlled drilling and riveting systems. Electroimpact Inc. supplied British Aerospace Airbus with a wing panel assembly cell for the new A340-500/-600 aircraft.

The cell is installed at the company's Airbus wing manufacturing and final assembly facility in Broughton. The cell incorporates two E4100 wing-riveting machines for upper and lower panel assembly. E4000 wing cell has significantly improved British Aerospace's manufacturing process. E4000 combines and automates three separate processes: attaching the wing skins to their supports (stringers); drilling them; and riveting them.

E4000 tacks, rivets, and bolts the wing skin to the stringer in one operation (Figure 33) is according to the correct wing box connections Fig.34.

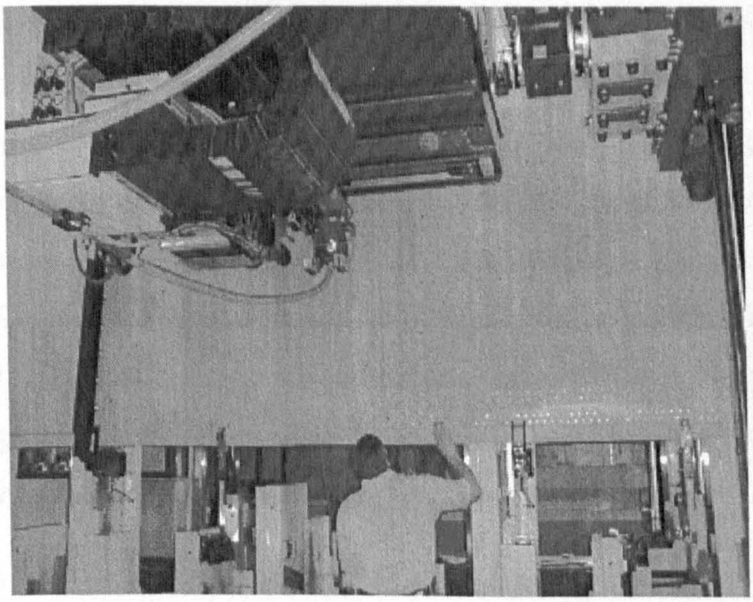


Figure 33: Installing Rivets and Boltlocks on A340 wing panel (© Electroimpact Inc)

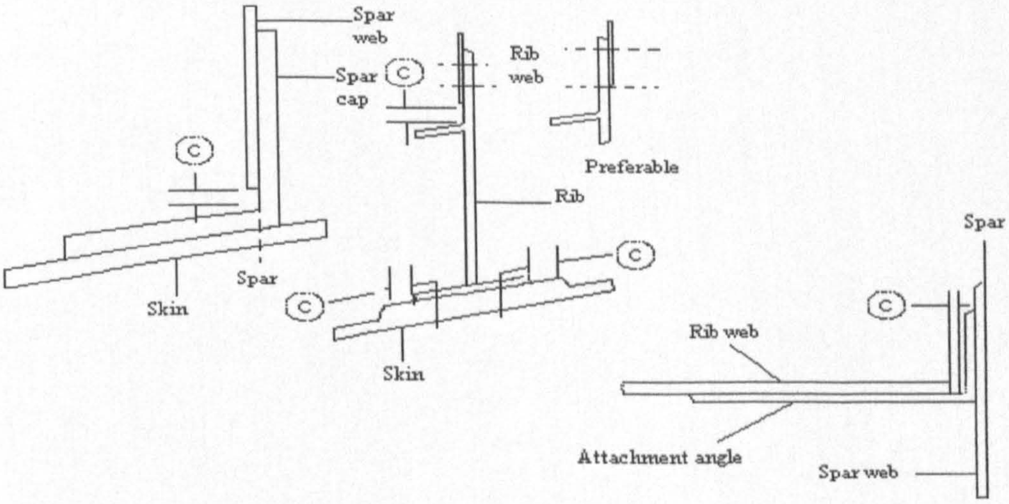


Figure 34: Correct wing box connection [Niu,1999]

Electroimpact provided six automated wing panel-holding fixtures, three for upper panels and three for lower panels. Each fixture has the flexibility to hold both port and starboard segments of the panel.

The IDMCS provides the mismatch control for assembly process using initial technological information from Electroimpact and engineering public available data from Airbus. The development of IDMCS for this case study is described in details in next section of dissertation.

8.2 Ontology of knowledge. Implementation IDMCS for Wing Box Virtual Mock-up

The realisation process combines the steps necessary to create a generic ZEUS agent with the steps necessary to implement the role-specific solutions identified during the previous phase. It is decided to create several agents to fulfil the roles found within the role model:

Table 10: Tasks definitions

Name Details	**Roles Played**
Rib	Assembly
Stringer	Assembly
Bottom skin	Assembly
Boltlock	Assembly
Nut	Assembly
WingBox	MakeWingBox

The Table 11 represents a definition of agents and M_{par}^{cr}.

Table 11: Definitions of Agents

Name Details	M_{par}^{cr}	**Roles Played**
Rib	Hole_ Diam	Assembly(Schedule, Rib_i, $Stringer_i$)
Stringer	Hole_Diam	Assembly(Schedule, Rib_i, $Stringer_i$, Bottom skin)
Bottom Skin	Hole_Skin_Diam	Assembly(Schedule, Rib_i, $Stringer_i$, BottomSkin)

Name Details	**M_{par}^{cr}**	**Roles Played**
Boltlock	Bolt_diam_ Meansured Bolt_length_Meansured Material Code Head Type	Assembly(Schedule, Rib_i, $Stringer_i$, Bottom Skin)
Nut	Nut_type Material Code	Assembly(Schedule, $Boltlock_i$)
WinBox	-	MakeWingBox (Boolean - True or False)
ANS	-	Agent Name Server
Broker (C-agent)	-	Broker (Faciliator)
Visual	-	Visualiser

Each role played by an agent entails some responsibilities, e.g. resources that will need to be produced or consumed, interactions with external systems etc.

Hence the next stage is to use the role descriptions to create a list of responsibilities for each agent.

The design process is a process of refinement, mapping each of the responsibilities identified in the previous stage to a generalised problem, and then choosing the most appropriate solution.

The responsibilities involved can be categorised as social or domain responsibilities, the former involving interaction with other agents, and the latter involving some local application-specific activity; this results in the following:

Table 12: D-Agent. Social Responsibility

Agent	**Social Responsibility**
D-agent	To send the critical parameters(M_{par}^{cr}) information
	To request M_{par}^{cr} from other D-agent (defined by schedule)
	To receive information from D-agents
	To send information to C-agents

Table 13: D-Agent. Domain Responsibility

Agent	**Domain Responsibility**
D-agent	To analyse an input information from D-agents
	To modify of design project
	To adapt a knowledge-base
	To adapt a fact base

The next role to consider for C-agent:

Table 14: C-Agent. Social Responsibility

Agent	**Social Responsibility**
C-agent	To request design situation information from D-agents
	To receive design situation information from D-agents
	To send a schedule information to D-agents
	To send requested information to other C-agents
	To request information from C-agents
	To adapt of knowledge-base of D-agents
	To adapt of fact-base of D-agents

Table 15: C-Agent. Domain Responsibility

Agent	**Domain Responsibility**
C-agent	To analyse an input information from D-agents
	To analyse information from C-agents
	To adapt of meta-knowledge base
	To adapt a fact-base

Now we have a list of the responsibilities for each intended agent. The design process can commence.

Table 16 shows external programs needed for the organisation of Designer- IDMCS negotiation (VM program) and organisation interface between PARASOLID and IDMCS for transmitting and receiving a geometric data.

Table 16: External Programs

Program	**Description**
VM	Virtual Mock-up General Configuration Interface
PARASOLID	Interface with PARASOLID Kernel

8.3 IDMCS software development and the results

The system is designed using JAVA 1.2.1 in the Windows NT environment using ZEUS agents building toolkit. The Figure 35 shows the Agent Society for wing assembly process.

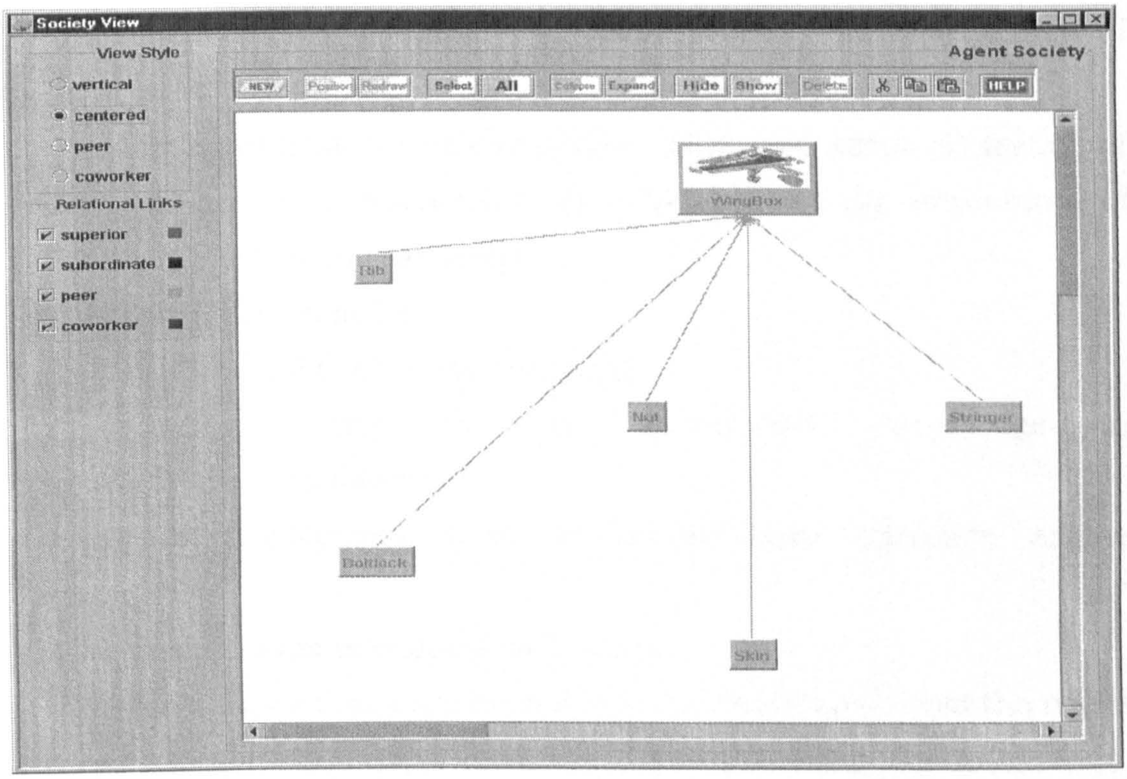
The result of IDMCS implementation is PARASOLID visualisation and agent internal representation. At this stage IDMCS will detect the tolerance mismatches.

8.3.1 Integration of IDMCS and Parasolid Kernel. Developing of external Interface

The principles of integration of ZEUS and Parasolid were described in section 6.3. In this part we will only define the information flows between these systems.

This stage of research is oriented towards developing tools and toolsets for IDMC. The organisation of integration between distributed knowledge-base based on ZEUS and PARASOLID to transfer CAD information directly to ZEUS created agents and to receive output from IDMCS.

At this stage it is important to organise a correct data transfer through external interface without loss of information which is important for design.



View Style
○ vertical
● centered
○ peer
○ coworker

Relational Links
☑ superior
☑ subordinate
☑ peer
☑ coworker

Figure 35: ZEUS Toolkit. Wing box design

In next sub-chapter the implementation of IDMC-approach for distributed design is outlined.

8.3.2 Implementation of IDMCS for aerospace design and evaluation of results

Implementation of IDMCS for aerospace design and evaluation of results is one of the important stages of this research.

The testing of IDMCS was carried out fall in two parts: (1) testing of IDMCS software and distributed knowledge-base, (2) evaluation of IDMCS results based on real design.

The first stage includes:

1) The quality of testing examples;
2) The correctness of distributed knowledge-base (completeness, consistency);
3) The effectiveness of knowledge base inference engine strategies.

The second stage is realised as follows:

The evaluations that have been conducted have supported the needs for three different strands of research: (1) design time, (2) design cost, and (3) raise designer level.

At this stage a set of experiments were planned. The first stage was to implement extended prototype for number of design situation using current data about A340 wing design construction. The second stage is an experiment when IDMC has been used to evaluate results design of TUPOLEV Corporation and AVIASTAR. The third stage was about implementation of IDMC-approach for organisation of distributed design environment within Aviation Euro-Russian Consortium.

The evaluation of the results of implementation of IDMCS was combination of objective characteristics as reducing design time, cost of design and characteristic based on subjective factors: (1) conveniences, (2) usefulness, and (3) informativeness.

The design experiment conducted is represented in Table 17.

Table 17. Plan of experiments completed

Experiment	**Number of elements in assembly**	**Parameters For testing**	**Data for Design**	**Evaluation of distributed knowledge-base**	**CE support**
A340 Wing box assembly	≤100	Initial testing	Technical documentation, aircraft Structural design books, Observation	Yes, according to knowledge engineering issues and data received Inc.	Design for assembly (DFA)
TU-204C, AVIASTAR	≤100	Initial testing	Parts and assemblies data catalogue, the data received from AVIASTAR designers	No	DFA
TU-324, TUPOLEV Corporation	≤100	Formal parameters: Reducing time of design Subjective parameters: conveniences, usefulness, informativeness	TUPOLEV Corporation, project data information	Yes, according to designers knowledge TUPOLEV Corp.	DFA Design for Manufactu-rability (DFM)

Experiment	**Number of elements in assembly**	**Parameters For testing**	**Data for Design**	**Evaluation of distributed knowledge-base**	**CE support**
Aviation Euro-Russian Consortium (AERC)	< 200	Organisation of Distributed Mismatch Control Process	AERC partners	TUPOLEV AVIASTAR	Assembly Manufactu-rability

The next section reviews the structure of distributed design and mismatch control environment within Aviation Euro-Russian Consortium.

8.4 IDMC-approach for distributed collaborative design

8.4.1 An example of distributed design environment for Consortium based projects

This part describes the implementation of IDMC-approach for distributed design process and the role of IDMCS in detection and handling of design mismatches.

The situation when, two companies in a consortium building components that don't fit together, with neither being able to tell the other to change the product, so that agents can negotiate a solution. We are tried to find evidence for this scenario happening in real life and Aviation Euro-Russian Consortium is a good example of this scenario.

An example is based on collaboration of two main partners with design and manufacture infrastructure, but using the different design and drawing standards, software, and design methods and typical solutions. The main idea is to integrate a design and manufacture capability of consortium partners to reduce the cost of production and design, as well as to improve the time of development of new products.

We have try to argue, why intelligent agents are better in CSCW for some purposes than having general collaboration design environment, which would allow people to do face to face negotiations. In this situation the language barrier one of the main problems. Moreover the cultural differences [Mammersley and Atkinson, 1996] in handing mismatches is another problem. There is the old argument, that human (designer) often prefer criticism from machines to criticism from humans.

The integrated structure of Consortium based on coordinated company which analyses the information from partners and manually (based on design consultants skills) working to reduce the inconsistencies of design and manufacture process. The structure is represented on next figure (Fig. 36).

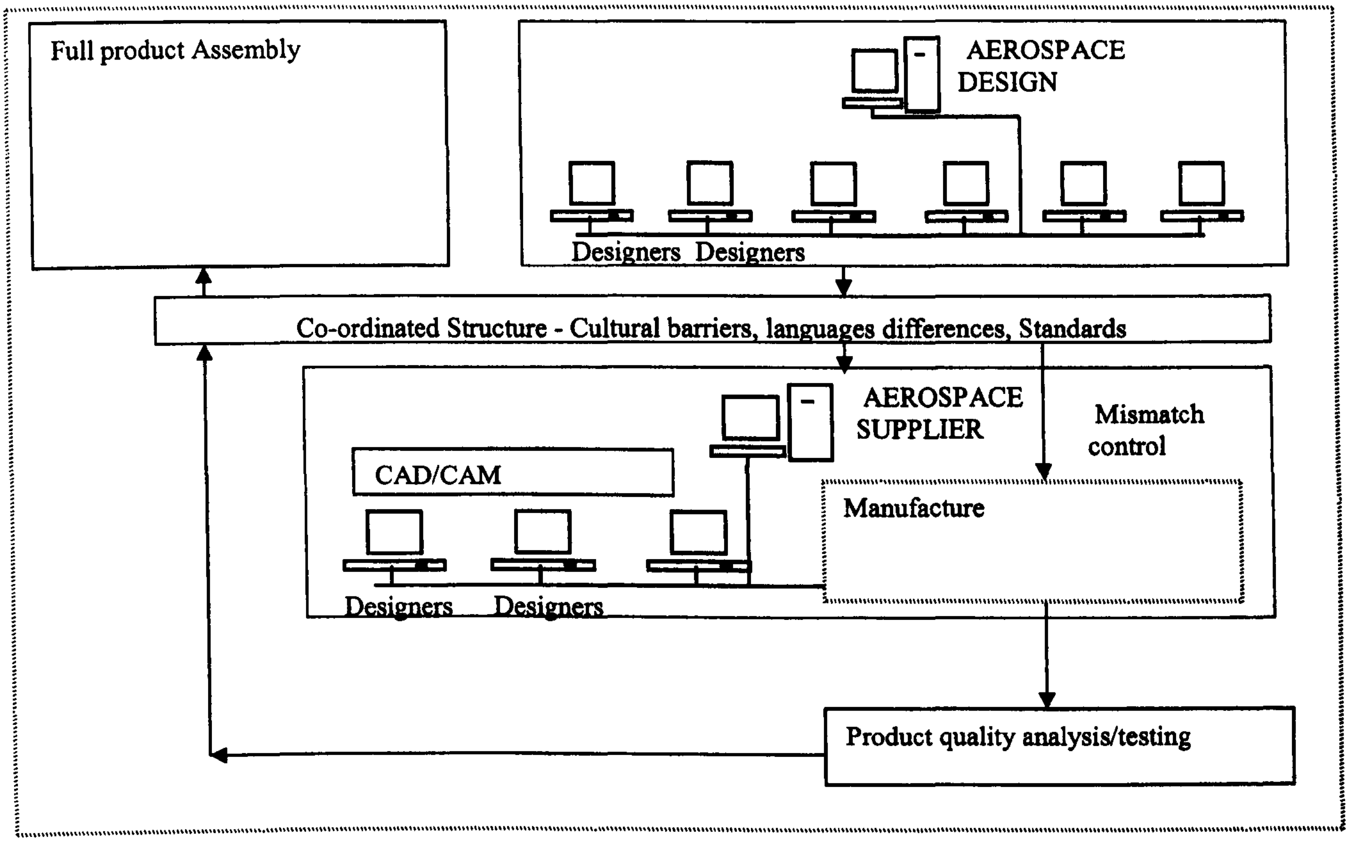


Figure 36: The Aerospace Consortium structure

As you can see from the figure, there are no direct contacts between designers at the different consortium partners. Also he process of product testing and control is distributed but not really effective. The idea is to change infrastructure, to reorganise the relations between designers and design management.

The IDMCS can help to establish this missing link. The improved infrastructure and IDMCS system in this communication is presented below:

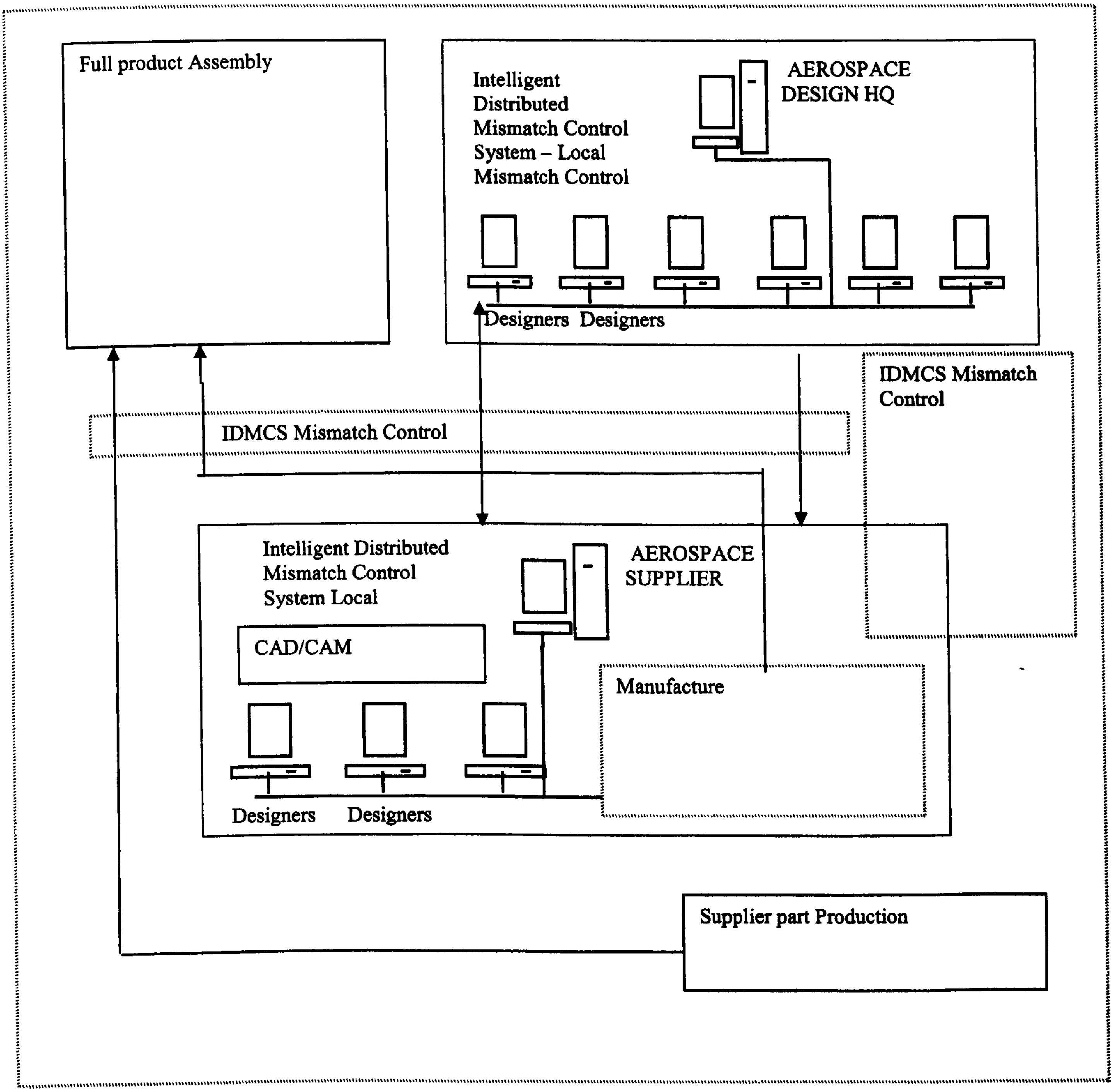


Figure 37: Implementation of IDMC for Consortium Based Design

The prototype of IDMCS also was used for design of elements of fuselage of TU-204C TUPOLEV (Fig. 40, Appendix D) commercial aircraft for AVIASTAR Company (Ulyanovsk Aerospace Industrial Complex), and TUPOLEV Corp. for design of elements TU-342 (Fig. 39, Appendix D) business jet plane. The details about industrial collaborators are represented also in Appendix D of the dissertation.

8.5 Conclusion

In this part the case study about the implementation of IDMCS for aerospace related application (wing-box design) was described.

The ontology of C- and D-agent agents for development of IDMCS for wing-box mismatch control was introduced.

This case study review the successful implementation of IDMCS for real–life problem, and describes how IDMCS can deals with mismatches of detail design stage of design process and to help to organise the unified infrastructure for distributed design/manufacture/assembly.

We have examined a number of concrete examples of mismatch handling at our industrial partners such as TUPOLEV, AVIASTAR, mapped the mismatch handling process, and conducted the series of interviews.

We have found what models, sketches, and other information representations are involved in communication and joint problem solving.

Finally, the series of experiments were conducted and environment for Distributed Collaborative Mismatch Control for Aviation Euro-Russian Consortium was developed.

Chapter 9 Summary and Conclusion

This section pulls together the work, which has been done in this project. Section 9.1 summarises the research findings and deliverables achieved. This section also contains a few comments by the author on some of the qualitative aspects of the work that could not be isolated to any single preceding chapter. Section 9.2. looks at the conclusions reached and how the IDMC project area can be extended.

9.1 Summary of research findings. Development of IDMC-approach. Progress to date

The main aim of this research was to develop a methodology, models and tools for detecting design inconsistencies in a distributed design environment.

The title of this thesis: 'A multi-agent approach to design consistency checking' was chosen to reflect the possibility of implementing a Distributed AI framework, based on multi-agent systems, as an effective approach to consistency checking in distributed environments.

How effective is a multi-agent approach to design consistency checking, especially for distributed design of complex systems?

This is the research question adopted in this thesis. The research was based on the hypothesis that distributed AI, particularly multi-agent systems, can be effective for consistency checking. This is particularly true in the case of complex products.

We have successfully used a research methodology that has been conducted along four main stages/phases of research (Section 1.4).

These phases are: Phase 1 (Initial Phase): Hypothesis definition, Literature review, Initial data collection; Phase 2 (Model development): The definitions of general model of mismatch control process, general principles of mismatch control, general taxonomy of design mismatches, multi-agent structural and functional framework; Phase 3 (Research Prototype development): Developing of the research prototype, Initial testing of research prototype; and finally Phase 4 (Industrial Application): The implementation of a theoretical framework for a specific Industrial case study.

Summary of Contributions

In this thesis, I introduced the Intelligent Distributed Mismatch Control approach (IDMC-approach), and then showed how IDMC can be implemented for developing an IDMC-based system: the IDMCS. IDMC was outlined in Chapter 4. The general methodology was represented as comprising two sub-models: process model of IDMC and structural model - conceptual multi-agent framework.

The conceptual framework for the development of a taxonomy of mismatches was represented, as well as the implementation of this framework for a DFA/DFM taxonomy.

Development of IDMCS was outlined as well as the possibility of using IDMCS for aerospace design.

The research question stated in Chapter 1 has been solved: as shown in Chapters 4 to 8, the IDMC-approach was introduced and implemented. As a result, the main contributions to knowledge of this thesis include:

1. The comprehensive research report outlining literature, previous work in this field, methods, applications, possible ways for investigations. This helped us to understand the complexity of mismatch control process, the gap between existing methods and tools for consistency checking. The report also presents Industry needs.

2. IDMC-approach. The Methodology of Distributed Mismatch Detection in Design. Process Model of IDMC.

3. Taxonomy of design mismatches.

4. Conceptual Framework required enabling the co-operating agents to detect mismatches and decide how to resolve them.

5. The principles of organisation and development of IDMCS.

6. The principles of Mismatch Control in Aerospace Design. The methods of conflict resolution in aerospace design using IDMC-approach.

7. The research prototype based on the new MAS Framework including practical designer's knowledge (i.e. real design situations, facts and rules for the mismatch detection and correction).

We can see that, these contributions reflect on the overall aim of this research project. Also, this thesis was an important step forward to discover the complexity of mismatch control process and how important for engineering design to investigate this area further.

The research methodology used was fully valuable during this research, to investigate different 'what-if' scenarios, for instance, for multi-agent model. For example, the dynamic automata model of intelligent mismatch control is an important deliverable of this project.

It was easy to modify this model quickly to perform actions in a different order, or to impose additional restrictions to judge the change in behaviour for multi-agent framework.

The next section will discuss possible future developments of this project.

9.2 Future work. Intelligent distributed mismatch control as a way for a new design approach

Research results introduced in this thesis provide a valuable and practical tool to use for design of intelligent distributed mismatch control systems.

The long term goal of this work will be to provide a future solid foundation for the development of distributed mismatch control systems. Previous development of consistency checking models has either been from scratch, or as in the case of some approaches, based on a previous system developed at the same site.

The next logical progression will be the extensive testing and analysis of Industrial implementation of IDMC-approach.

This is a major task in its own right, but one which requires a solid foundation. This foundation must provide both a mechanism for performing a comparative analysis, as well as an evaluation of existing approaches to prevent duplication of effort. It is this foundation that the work of this thesis is intended to provide.

This work has established both the needs for continuing research in the area of IDMC, and the areas within which further work is required. The possible future extensions are described below:

With the work described in this thesis as the foundation, it is now feasible to tackle this next set of exciting challenges.

IDMC-approach. Automata Model

This stage of investigation was oriented towards defining algorithms and strategies for mismatch resolution. There remain lots of questions about knowledge organisation, rules types, adaptation algorithms, which will provide the best designer support during mismatch control process.

It is proposed that further, more in-depth analysis be performed to identify the formal representation for best description of dynamics of a multi-agent systems, for instance Petri-nets, Colored and modified Petri-nets, DEVS-representation, etc.

Agent Communication Language

One thing, which currently prevents the implementation of advanced types of agents (including learning agents), is the restrictive nature of FIPA agent communication languages. There is a need for more research to define an extension to ACL syntax that would support a mismatch control capability. Time has not allowed for such a topic to be investigated in details here, but various approaches for agent-based CE systems were described in Chapter 2.

Intelligent Distributed Mismatch Control System

It is intended that the future work will further refine and expand upon the model of Intelligent Distributed Mismatch Control System.

This work will be broken into two inter-related research areas: (1) Extended Prototype development and (2) Commercial implementation of IDMCS. All these areas will be based upon an extension of the system performance in dealing with mismatch control.

Overlapping these research areas are topics such as:

- development of an advanced library of taxonomies of design mismatches for different industrial applications;
- extended knowledge representation for aerospace industry and formation of commercial kernel of the system;
- implementation of learning algorithms based on Soft Computing techniques.
- commercial implementation of IDMCS.

Conflict Management in Virtual enterprises

Virtual enterprises [Arnold, et al, 1995; Nayak et al, 2001; Xu et al, 2002; Umar and and Missier, 1999] are a future development of traditional enterprises and based on future distribution of processes, tasks, supply chains, ... etc.

The IDMC approach can be used to investigate the possibility of improving communication within virtual enterprises through the use of multi-agent framework.

These models can be refined as the nature of the communication in these systems becomes better understood.

The changing nature of modern enterprises can also be expected to influence the nature of distributed virtual enterprises. The approaches underlying these systems can be expected to evolve to meet these changing requirements.

Ethnographical aspects of IDMCS Implementation

Another area is ethnography or cultural aspects [Mammersley and Atkinson, 1996] of the mismatch control process.

In Section 8.4. some aspects were briefly described, using the collaboration example of two main partners with design and manufacture infrastructure, but using different design and drawing standards, different software, and different design methods. The cultural and social differences between partners can raise the number of design mismatches dramatically. More research is required in this area, including interdisciplinary projects involving psychologists and sociologists.

Epilogue

I believe that this thesis is a beginning, of an extensive research related to understanding the mismatch control process in engineering design. I hope that the methodology and implementation described in this thesis will lead to a future development of a commercial computer supported collaborative design and concurrent engineering tools, especially for Aerospace related applications.

REFERENCES

Automatic design Verification, ADV V3 (1999), Tecoplan AG

AGENTS Kernel Language, AGENTS Kernel Language User Manual: http://www.sics.se/isl/akl/

AIDA: Artificial Intelligence supported Design of Aircraft: http://www.kbs.twi.tudelft.nl/Research/Pr ojects/AIDA/

AIT Initiative: Advanced Information Technology for European Manufacturing Industry: http://www.ait.org.uk

Addressing the CAD/CAM/CAE Interoperability Issue (1999), *White Papers, Unigraphics Solutions*, 1999.

Akman, V., Hagen, P. J. ten, and Tomiyama, T. (1990). A Fundamental and Theoretical Framework for an Intelligent CAD System. *Computer Aided Design Journal, 22, pp.352-367.*

Akman, V., ten Hagen, P., Rogier, J., Veerkamp, P. (1987) Knowledge engineering in design, Centre for Informatics and Computer Science CWI Amsterdam, *Report CS-R8745.*

All set for more wings, Technology update (1999), Aerospace Engineering Online:

http://www.sae.org/aeromag/techupdate/06-1999/06.htm

Appelbaum., S., Abdallah., C., Shapiro., B. (1999). The self-directed team. A conflict resolution analysis. *Team Preformance management. 5 (2), 60-77.*

Arnold, O., Faisst, W., Hartling, M. & Sieber, P (1995) *Virtuelle Unternehmen als Unternehmenstyp der Kukunft?* In: Handbuch der modernen Datenverarbeitung – Theorie und Praxis der Wirtschaftsinformatik, Volume 185. Heidelberg: Huthig-Verlag, 1995. Cited in: Fischer, K, Muller, J, Heirnig & Scheer, A (1996), *Intelligent Agents in Virtual Enterprises.* In Proc 1st International Conference on Practical Applications of Intelligent Agents and Multi-Agent Technology (PAAM), London, 1996

Automatic design Verification, AVD V3 (1999). Tecoplan AG, Germany, 1999

Automated wing box assembly: Aerospace Engineering Online (1999): http://www.sae.org/aeromag/techupdate_10-99/03.htm

Banis, D., Marceau, A. et. al. (2000). *Design for Corrosion Control, Aero, 7.*

Bechkoum, K. (1997). Intelligent Electronic Mock-up for Concurrent Design, *Expert Systems with Applications Journal,* 12, pp. 21-36.

Bechkoum, K., Taratoukhine, V., (1999a). Software Agents to Support Geometric Modelling., Expert Update (BCS SGES), June 1999.

Bechkoum, K., Taratoukhine, V., (1999b). A Framework for Mismatch Control in a Distributed Design Environment., Proc. Advances in Concurrent Engineering, Bath, 1-3 September 1999.

Belov, Y. (1989). New paradigms of simulation techniques application in computer - aided design. *System Analist, Modelling, Simulation* 6 (8), 629-631.

Bennett, G. (1997). The application of virtual prototyping in the development of complex aerospace products. *Aerospace engineering and Aerospace Technology. 69 (1), pp 19-27.*

Bento, J., and Feijo, B. (1997). An Agent Based Paradigm for Building Intelligent CAD Systems. *Artificial*

Intelligence in Engineering Journal. 11, pp. 231-244.

Bento, J., (2000) Web-Based Design: do we need to formalize it?, in The Lisbon Charrette: A Collaborative Design Studio, IST Press, Lisboa.

Bento, J., (1998), Modelling Mechanical behaviour without mechanics in Development of Knowledge-Based Systems for Engineering, C.Tasso and E. Arantes e Oliveira(eds.), 37-58, Springer-Verlag.

Brazier, F., Eck, P., Treur J. (1989). Design of a modelling Framework for Multi-agent systems, *Trends in Theoretical Informatics Band 89 in Schriftenreihe der Osterreichischer Computer Gessellschft*, Wien, Munchen, *pp. 173-191.*

Brazier, F., Jonker, C., and Treur, J. (1996). Modelling Project Coordination in a Multi-Agent Framework, Proceedings of the 5th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE '96).

Brooks, R. (1990). Intelligence without Representation, MIT, Research Report.

Brown, D., Grecu, D. (1997). An Al in Design View of Design, *Technical Report*, CS Dept., WPI.

Brown, D., Grecu, D. (1996). Dimensions of Learning in Agent Based Design, 4th International Conference on AI in Design - *Workshop on Machine Learning in Design*, Stanford, CA, June 24-27.

Bobrowski, R., Biddulph, M. et al. (1999). Achieving concurrent engineering objectives through the large-scale use of computer-based models of engineering design process, *Proc. Advances in Concurrent Engineering*, Bath, 1-3 September.

Bond, A., and Ricci, R., (1992). *Cooperation in aircraft design.* Research in Engineering Design, *pp.115-130.*

Borg, J., Yan X. et. al. (2000) Exploring Decisions Influence on Life-Cycle Performance To Aid Design For Multi-X International Journal *Artificial Intelligence in Engineering Design & Manufacture, 14, pp. 91-113.*

Bradford, J. (1995). Full-circle appraisal at BAe, Management Development Review. 18 (4), *pp. 10-13.*

Bucciarelli, L. (1994). Designing Engineers, MIT Press.

Butler, R. (1998). Optimisation wing structures – theory or practice, *Aerospace engineering and Aerospace Technology. 70 (1), pp. 4-8.*

Buyer's Guide. (1999). Mechanical CAD, *CAD/CAM Magazine. 18 (10), pp. 58-62.*

Castelfranchi, C. (1996). Conflict Ontology, in Proceedings of ECAI 96, 12th European Conference on Artificial Intelligence.

Cengeloglu, Y. (1995). AGENT_CLIPS Version 1. 0.

CATIA, DMU System, IBM: http://www.catia.ibm.com/catmain.html

CERL project. Conflict management in collaborative design: http://impact.usc.edu/cerl/

Cointe, C., Matta, N., Ribire, M. (1997). Design Propositions Evaluation: Using Viewpoint to manage Conflicts in CREoPS 2, *In Proceedings of ISPE/CE,* Concurrent Engineering: Research and Applications, Rochester August.

CORBA, Common Object Broker Architecture: http://www.omg.org/corba/

Cost, R., Chen, Y., Finin, T., et al. (1999). *Modeling agent conversations with Colored Petri Nets,* Third Conference on Autonomous Agents, Workshop on Agent Conversation Policies, Seattle, WA.

Cutkosky, M., Mori, T. (1998). Agent-based collaborative design of parts in assembly, Proceedings of DETC'98 1998 ASME Design Engineering Technical Conferences September 13-16, Atlanta, Georgia, USA.

Cutkosky, M., Engelmore, R., Fikes, R., Genesereth, M., Gruber T. et. (1993). AI PACT: An experiment in Integrating Concurrent Engineering Systems, *IEEE Computer* 26(1), 28-37.

D'Ambrosio, J., Darr, T., and Birmingham, W. (1996). Hierarchical Concurrent Engineering in a Multiagent Framework, *Concurrent Engineering: Research and Applications Journal.* 4, pp. 47-57.

Daberkow, D., and Marvis, D. (1998). New Approaches to Conceptual and Preliminary Aircraft Design: A Comparative Assessment of a Neural Network Formulation and a Response Surface Methodology, *Proceedings of World*

Aviation Congress and Exposition, Anaheim, CA, USA, Paper N. SAE-985509.

Darr, T., Birmingham, W. An attribute-space representation and algorithm for concurrent engineering (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing, vol.10(1), 21-36.

Davidow, W., Malone, M. (1992). The virtual corporation: structuring and revitalizing the corporation for the 21st century. - N.Y.: Harper Nollins.

Design Process in the Aerospace Industry, PSRC Innovative Manufacturing Initiative: http://www.dig.bris.ac.uk/aero/welcome.htm

Duffy, A., Andreasen, M. (1999). Design Co-ordination, *International Conference on Engineering Design (ICED'99)*, Munich, Germany, 24-26 August.

Dunskus, B., Grecu, D., Brown, D., Berker, I. (1995). Using single function agents to investigate conflict (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 9(4), p. 299-312.

Easterbrook, S. (1991). Handling conflict between domain descriptions with computer-supported negotiation, Knowledge Acquisition, 3, pp. 255-289.

Easterbrook, S., Beck, E. et. al. (1993). A Survey of Empirical Studies of Conflict , in CSCW: Cooperation or Conflictp Edited by S. Easterbrook, pp. 1-68.

Easterbrook, S., et. al. (1994). Coordinating Distributed ViewPoints: The Anatomy of a Consistency Check, International Journal on Concurrent Engineering: Research and Applications, 2(3), pp. 209-222.

Favela, J., Wong, A., and Chakravarthy, A. (1993). Supporting collaborative engineering design Engineering with computers. 9, pp.125 - 132.

Ferber, J. (1999). Multi-agent systems. An Introduction to Distributed Artificial Intelligence, Addisson-Wesley Longman.

Fernandes, F., and Belo, O. (1998). Modeling Multi-Agent Systems Activities Through Colored Petri Nets: An Industrial Production System Case Study. 1*6th IASTED International Conference on*

Applied Infomatics (AI'98), Garmisch-Partenkirchen, Germany, pp. 17-20.

Fischer, K. (1994) The Design of an Intelligent Manufacturing System , Proceedings of the 2nd Intl.~Working Conference on Cooperating Knowledge-based Systems (CKBS'94) (Selected Papers), Deen, S. M. (ed.), Dake Centre, University of Keele, 1994

Foddy, W. (1995). Constructing questions for interviews and questionnaires, Cambridge University Press, Cambridge

Ford, T. (1998). Continuing wing production Aircraft Engineering and Aerospace Technology. 70(1), pp. 9-14.

Frost, H., and Cutkosky, M. (1996). Design for Manufacturability via Agent Interaction, *Proceedings of ASME Design for Manufacturing Conference,* Irvine, CA, August 18-22, Paper N 96-DETC/DFM-1302.

Ganesham, R., Prakash S. (1996) Collaborative Design Systems. An annotated bibliography/ CERL survey project.
www.cecer.army.mil/pl/ace/survey.htm

Gascoigne, B. (1995). PDM: the essential technology for concurrent engineering, World Class Design for Manufacture. 2(1), pp. 38-42.

Gero, J. (1998). Adaptive Systems in Designing: New Analogies from Genetics and Developmental Biology, in *Adaptive Computing in Design and Manufacture*, I. Parmee (ed.), Springer, London, pp. 3-12.

Gohen, G., Moller, J., Quadrat, M. (1989). Algebraic tools for the performance evaluation of discrete event systems, Proceedings of IEEE. 77, pp. 39-58.

Gorodetski, V., Lebedev, A. (1998). Multi-agent Technology for Planning, Scheduling and Resource Allocation. *Proceedings of the Third International Conference on Multi-agent systems*. IEEE Computer Society, pp.429-430.

Grasso, F. (1998). *When No-Conflict Means No-Collaboration - advice giving in the health promotion domain,* Working Notes of the ECAI 98 Workshop: Conflicts Among Agents: avoid or use themp Brighton, UK, 25 August, p. 69-75.

Gupta, L., Chionglo, J., and Fox, M. (1996). A Constraint-based Model of Communication and Coordination in Concurrent Design Projects, *Proceedings of the 5th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE '96).*

Hale, M., and Craig, J. (1994). Preliminary Development of Agent Technologies for a Design Integration Framework, *Proceedings of 5th Symposium on Multi-disciplinary Analysis and Optimisation,* Panama City, FL, USA, September 7-9.

Hale, M. (1994). IMAGE: A Design Integration Framework Applied to the High Speed Civil Transport, 1st Industry/Academy Symposium on Research for Future Supersonic and Hypersonic Vehicles, Greensboro, NC, December 4-6, HM301.

Hammersley, M., Atkinson, P. (1996). Ethnography, Principles in Practice, London, Tavistock Publications

Hale, M., Craig, J. (1995). Use of Agents to Implement an Integrated Computing Environment, *Proceedings of AAAI Conference Computing in Aerospace,* San Antonio, TX, March 28-30.

Hale, M., Craig, J., et. al. (1996). DREAMS and IMAGE: A Model and Computer Implementation for Concurrent, Life-Cycle Design of Complex Systems, Concurrent Engineering: Research and Applications. 4(2), June.

Hardwick, M., Spooner D., Rando T., Morris K. (1996). Sharing manufacturing information in virtual enterprises. *Communications of the ACM.* 39(2), pp.46-54.

Hill, K. (1997). Corrosion prevention in the Royal Air Force: there is no option, Anti-Corrosion Methods and Materials. 44(2), pp.107-114.

Holland, J. (1968). Processing and processors for Schemata, In *Associative Information Processing,* American Elsevier, pp.127-146.

Holvoet, T. (1995). *Agents and Petri Nets* . In the Petri Net Newsletter, No. 49,. Publ. Gesellschaft fuer Informatik, Bonn, Germany. Ed. Joerg Desel. Cover picture story, 3-8. Abstract available. Postscript version available.

Hopcroft, J., Ullman J. (1979). *Introduction to Automata Theory, Languages and Computation,* Hardcover, 418 pages, Addison-Wesley Pub Co.

Hyacinth, S., Nwana, H. (1998). ZEUS: An Advanced Tool-Kit for Engineering Distributed Multi-Agent Systems, Divine T. Ndumu & Lyndon C. Lee. *Proceedings of PAAM'98*, London, March, pp.377-392

IDEAS: http://www.sdrc.com/nav/software-services/ideas/

Inan K., Varaiya, P. (1989). Algebras of discrete event models. Proceedings of IEEE. 77, pp.24-38.

Jacquel, D., Salmon, J., Mill, F. (1997). Features as Agents, An Alternative Paradigm for Concurrent Engineering , ICMA'97 28-30 April,Hong-Kong.

Jennings, N., Sycara, K., Woldridge, M. (1998). A roadmap for agent research and development, *Autonomous agents and multi-agent systems* N1, Vol. 1.

Jin, Y., et al., (2001) An agent-based Decision Network for Concurrent Engineering Design, Concurrent Engineering: Research and Applications-CERA Journal, Vol 9., N 1, March 2001

Joslyn, C., Rocha, L. (1999) Towards Semiotic Agent-Based Models of Socio-Technical Organizations (LAUR 99-5670, 00-350) presented at the *2000 Conference on AI and Simulation AI, Simulation and Planning AIS'* 2000 March 6-8, 2000 Tucson, Arizona, USA.

Kaelbling, L., Littman, M., Moore, A. (1995). Introduction to Reinforcement Learning, in The Biology and Technology of Intelligent Autonomous Agents (Steels, L), *NATO ASI Series*, Springer. 144, pp. 90-127.

Kaelbling, L., Rosenschein, S. (1990). Action and Planning in Embedded Agents, *Robotics and Autonomous Systems*. 6, 35-48

Katragadda, P. (1997). The ICAD System – A Knowledge Based Generative Technology, *Proceedings of the Knowledge-based systems interoperability Workshop,* 3-4 November , Maryland, USA.

Kim, S. (1989). An Automata-theoretic Framework for Intelligent Systems, *Robotics and Computer Integrated Manufacturing*, Vol.5, No 1 pp.43-51.

Klein, M. (1991). *Supporting conflict resolution in cooperative design systems*. IEEE Transactions on Systems, Man, and Cybernetics 21(6). (Special Issue on Distributed AI).

Klein, M. (1995), iDCSS: Integrating Workflow, Conflict and Rationale-Based Concurrent Engineering Coordination Technologies , Concurrent Engineering: Research and Applications 3/1, pp. 21-27.

Klein, M. (1992). Supporting Conflict Management in Cooperative Design Teams, *Proceedings of the 11th International Workshop on DAI,* Glen Arbor, MI.

Knowledge Based Engineering at AIRBUS (1998) : British Aerospace AIRBUS Report.

Kock, N. (2000). Benefits for Virtual Organizations from Distributed Groups, Communications of the ACM, November 2000. 43(11).

Kolb, M. (1994). Object modeling with constraints for aerospace conceptual/preminary design, Knowledge-based Artificial Intelligence Systems in Aerospace, editors Burtine, Wray, et al, *SPIE Proceedings,* Vol 2244, 04/04-04/08/94, Orlando, FL, USA 315 pages pp. 172-183.

Kroo, I., Takai, M. (1988). A Quasi-Procedural, Knowledge-Based System for Aircraft Design, Stanford University Stanford, California *AIAA 88-6502* http://aero.stanford.edu/Reports/QPforDesign.html

Kuchar, J. and Yang, L. (1997). Survey of Conflict Detection and Resolution Modeling Methods , AIAA-97-3732, AIAA Guidance, Navigation, and Control Conference, New Orleans, LA, August 11-13.

Lander, S. (1994). Distributed Search and Conflict Management Among Reusable Heterogeneous Agents. PhD thesis, University of Massachusetts, February.

Lee, S., Kim, G., and Bekey, G. (1993). Combining Assembly Planning with Redesign: An Approach for More Effective DFA, *Proceedings of the 1993 IEEE International Conference on Robotics and Automation.* 3, pp. 319-325.

Lee, D., and Hahn, H. (1995). Virtual Assembly Production Analysis of Composite Aircraft Structures, Integrated Manufacturing Engineering Program for Advanced Transportation Systems, University of California.Los Angeles, California, ASME.

Lefever, D., Wood, K. (1996). Design for Assembly techniques in reverse engineering and design, *Proceedings* *of the ASME Design Engineering Technical Conferences and Design Theory and Methodology Conference*, August 18-22, Irvine, California.

Liening, A., Blount, G. (1998). Influences of KBE on the aircraft brake industry, *Aerospace engineering and Aerospace Technology*. 70(6), pp. 439-444.

Liu, J., and Sycara, K. (1994). Distributed Meeting Scheduling, *Proceedings of Sixteenth Annual Conference of the Cognitive Science Society*, 13-16 August, Atlanta, USA.

Lu, S., Udwadia, F. (1999). Conflict Management in Collaborative Engineering Design, A draft research progress report, Engineering Process Laboratories, U.S. Army Construction Engineering Research Laboratories, December. http://impact.usc.edu/cerl/termreport/termreport/may1999/May1999Report.html

Matta, N. (1996). Conflict Management in Concurrent Engineering: Modeling Guides, in Proceedings of ECAI 96, 12th European Conference on Artificial Intelligence.

Matta, N,, Cointe, C. (1997). *Concurrent Engineering and Conflict Management Guides*, Proceedings of ICED, Tampere, August.

McMahon, C., Browne, J. (1993). *Computer-aided Design and Manufacture*, Addison-Wesley Press.

Mohammad, M., Sayera J., Czuchry, A., Yasin, M. (1996). Aircraft engine overhaul system: a prototype for integrating multi-vendor platforms in a static computer-integrated environment, *Integrated Manufacturing Systems*, 7/3, 47-62.

Mukhopadhyay, S., Gupta A. (1998). Interfaces for resolving marketing, manufacturing and design conflicts. A conceptual framework, European Journal of Marketing. 32(1/2), 101-124.

Mullins, S., Anderson, D. (1998) Automatic identification of geometric constraints in mechanical assemblies, *Computer-Aided Design.* 30(9), 1998, pp.715-725.

Multi-Site Concurrent Engineering for Large Aircraft Engineering and Support, European Project Aerospatiale
http://www.avions.aerospatiale.fr/muscles

Nayak, N., Bhaskaran, K & Das, R. (2001). *Virtual Enterprises – Building Blocks for Dynamic e-Business,* Information Technology for Virtual Enterprises (ITVE) IEEE Proceedings, pp. 80-87

Ndumu, D., and Tah, J. (1998). Agents in Computer-Assisted Collaborative Design, *Artificial Intelligence in Structural Engineering* (LNAI No.1454), May, pp.249-270.

Niu, M. (1999) Airframe Structural Design : Practical Design Information and Data on Aircraft Structures, Technical Book Co

Nuseibeh B., Easterbrook S., Russo., A. (2000). Leveraging Inconsistency in Software Development, IEEE Computer, April.

Nwana, H., and Ndumu, D. (1999). Perspective on Software Agents Research, *The Knowledge Engineering Review,* January.

Nwana, H. (1996). Software Agents: An Overview, *The Knowledge Engineering Review Journal,* 11(3), November.

O'Leary, D., Kuokka, D., Plan R. (1997). Artificial Intelligence and Virtual Organisations, Communications of the ACM, January. 40(1).

Oliveira, A., Bento, J. (1996). On the Trend of Progress in Structural Engineering, in The Advances in Computational Mechanics, (book dedicated to Prof. Qian Lingxi in celebration of his 80th anniversary), Z.Wanxie, C. Gengdong, L.Xikui (ed.s), 99-105, International Academic Publishers, Beijing, China, 1996.

Paderis, C. (1996). An agent-based approach to the Design of Rapidly Deployable Fault Manipulators, *PhD thesis, Carnegie Mellon University,* Pittsburg.

Pan, Z., Zhang, M., Shi, J. (1997). Cooperation Management for Collaborative CAD systems, *Proceedings fo CAD and Graphics'97,* Fifth International Conference on CAD and CG December 2-5, Shenzhen, China, pp.622-627.

Paraphase PARASOLID (1995), Product Newsletter, Issue 12, September, 1995

PARASOLID General Information : www.parasolid.com

Petrie, C. (1996). Agent-Based Engineering, The Web, and Intelligence, *IEEE Expert*, December.

Petrie, C., Webster, T., Cutkosky, M. (1995). *Using Pareto Optimality to Coordinate Distributed Agents* , AIEDAM special issue on conflict management. 9, pp.269-281

Pham, D., Dimov, S. (1998). An approach for concurrent engineering, Proc Institute Mech Engineers. 212, Part B, ImechE.

Prasad, B. (1995). *Concurrent Engineering Fundamentals: Integrated Product and Process Organization*, Volume I Hardcover - 502 pages: Prentice Hall.

Quayle, M. (1999). Project Management in European Aerospace plc: a case study, Industrial Management & Data Systems 99/5, pp.221-226.

Regli, W. (1997). Internet-Enabled Computer-Aided Design, *IEEE Internet Computing*, January-February. 1(1), pp. 39-51.

Reithmaier, L., et al. (1991). *Standard Aircraft Handbook*, 232 pages, Tab Aero.

Richardson, J. (1999). Solid Performer - Solid Works 99. *CAD User*. 12(9), October, pp.14-25.

Rocha, L. (1999). Review of Agent Models: Encounters, Strategies, Learning, and Evolution (LAUR 99-5475) Delivered to PSL August.

Roy, R., Baker V., Griggs T. (2002) Full Service Supplier Cost Modelling , Proc. Advances in Concurrent Engineering, Cranfield.

Rzevski, G. (1998). Engineering design for the Next Millenium: The Challenge of Artificial Intelligence, The 86th Hawksley Memorial Lecture ImechE, 9/12.

Sehdev, K., Fan, I., et. al. (1995). Design for Manufacture in the Aerospace Extended Enterprise, *World Class Design to Manufacture*, 2(2), pp.28-33 MCB Press.

Schmitt, G. (1996). Multi-agent interaction in a complex, virtual design environment, Swiss Federal Institute of technology (ETH), Project 5003-039009, http://spp-ics.snf.ch/SPP-IF/5003-039009.html

Shen, W., Norrie, D. (1999). Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey, *Knowledge and Information Systems: An International Journal.*

Sriram, D., (1993) Computer Supported Collaborative Engineering, *AAAI Workshop, AI in Collaborative Design,* John S. Gero, Mary Lou Maher,and Fay Sudweeks, Cochairs, Technical Report WS-93-07, AAAI Press. 117-129

Smith, D. (1997) Strategic alliances in the aerospace industry: a case of Europe emerging or converging? European Business Review. 97(4), pp.171-178.

Solid Edge (1999). CAD/CAM Magazine. 18(10), pp. 37-42.

Sun, J., Zhang, Y., and Nee, C. (2000) Agent-based product design and planning for distributed concurrent engineering. Proceedings of the 2000 IEEE International Conference on Robotics and Automation. San Francisco, CA., USA

Subbu, R., Hocaoglu, C., and Sanderson, A. (1998). A Virtual Design Environment using Evolutionary Agents, *Proceedings of the 1998 IEEE International Conference on Robotics & Automation,* Belgium, pp. 247-253.

Sycara, K. (1998). Multiagent Systems, AI magazine. 19(2).

Szczerbicki, E. (1994). Distributed Multiagent system for concurrency evaluation: Conceptual Archtecture, SAMS.16, pp.277.

Taratoukhine, V., et al. (1997). Knowledge-based Network Simulation System, *Proceedings of the 7th International Fuzzy Systems Association Congress*, Czech Republic, Prague, pp. 532 – 537.

Taratoukhine, V., et al. (1997). Knowledge-Based LAN Simulation System, *Proceedings of The 2nd International SimTecT Conference 1997*, Canberra, Australia, 1997, pp. 35-38.

Taratoukhine, V., (2001a). An Agent-based methodology for consistency checking in engineering design, International Congress, AI in XXI Century, Gelendgik, Black Cost, Russia, 2001

Taratoukhine, V., et. al. (2001). Computer supported collaborative design: mismatch control, methodology and application, 8th International Workshop of the European Group for Structural Engineering Applications of Artificial Intelligence (EG-SEA-AI), UK, July, 2001

Taratoukhine, V., (2001b). Using distributed artificial intelligence for the support of the design process, 4th International Conference on Soft Computing and Measurements, SCM'2001, June 2001

Taratoukhine, V., Bechkoum, K., Stacey, M. (2001a). Virtual Design Team Management. Consistency checking process, New Models of business: managerial aspects and enabling technology, St. Petersburg State University and University of California, Berkeley, June, 2001

Taratoukhine, V., Bechkoum, K., Stacey, M. (2001b). Intelligent Distributed Mismatch Control for European Aerospace Industry, CAD/CAM/PDM - '2001, The International Conference and Exhibition, Institute of Control Sciences, Russian Academy of Sciences, Moscow, April, 2001

Taratoukhine, V., Bechkoum K., (2000a). An agent-based paradigm to support virtual design mock-ups, 14th European Conference on Artificial Intelligence, Workshop on Applied Semiotics, Berlin, Germany, 2000

Taratoukhine, V., Bechkoum, K., (2000b). Intelligent Distributed Mismatch Control, CSCMP-2000 Complex

Systems: Control and Modeling Problems, 19-24 June 2000 Samara, Russia

Taratoukhine, V., Bechkoum, K., (2000c).Intelligent Distributed Mismatch Control: Model and Application, 3rd International Conference on Soft Computing and Measurements, SCM'2000, June 27-30, St.Petersburg, Russia

Taratoukhine, V., (2000). Intelligent Detection of Mismatches in Distributed Design, Intelligent CAD International Conference, Journal of Taganrog Radiotechnical University, Vol 1, January 2000, Taganrog, Russia

Taratoukhine, V., Bechkoum, K., (1999a). Towards a Consistent Distributed Design: A Multi-Agent Approach, Proc Information Visualisation '99, London, IEEE Press, 1999.

Taratoukhine, V., Bechkoum, K., (1999b). Mismatch Detection in Distributed Computer Aided Design", Proc International Conf. On Computer Graphics & Vision, ACM SIGRAPH and Lomonosov Moscow University, Moscow, 26 August-1st Sep. 1999.

Taratoukhine, V., Bechkoum, K., (1999c).Mismatch Detection in Distributed Computer Aided Design", International Journal Computer Graphics and Geometry, Vol. 2, No. 3. St. Petersburg, Russia(reprint from Proc. International Conf. On Computer Graphics & Vision, ACM SIGRAPH)

Taratoukhine, V., Bechkoum, K. (1999d). Semiotic Multi-agent Framework for Intelligent Mismatch Detection in Distributed Design, Proc. 4th Int'l Workshop on Applied Semiotics and Semiotics Control ASC'99, Zelenograd, Russia, 53-54 pp.

Taratoukhine, V., Bechkoum, K. (1999e). A Multi-Agent Approach for Design Consistency Checking, Proc. International Conference Human Computer Interaction - 99, Ulyanovsk, Russia, 20-22 September 1999. 21-22 pp.

Telescript Language: http://www.genmagic.com

The European Aerospace Industry Meeting the Global challenge (1997). European Commission, 466, Brussels, 24 September 1997

The Python Programming Language: http://www.python.org

Tomiyama, T. (1993). Towards knowledge intensive intelligent CAD, *JSME-ASME Workshop on Design*, pp. 46-51.

Tomiyama, T., Kiriyama, T., and Umeda, Y. (1994). Towards knowledge intensive engineering, in *Knowledge Building and Knowledge Sharing*, IOS Press, pp. 308-316.

Tong G., Fitzgerald., B. (1994). Concurrent Engineering Battleground, World Class Design to Manufacture. 1(3), pp.45-48.

Torenbeek, E. (1982). *Synthesis of Subsonic Airplane Design*, Hardcover, 28 September, Kluwer Academic Publishers; ISBN: 9024727243.

Unan, R., and Dean, E. (1992). Elements of Designing for Cost, presented at *The* AIAA 1992 Aerospace Design Conference, Irvine CA, 3-6 February, AIAA-92-1057.

Umar, A and Missier, P. (1999). *A Framework for Analyzing Virtual Enterprise Infrastructure*, Research Issues on Data Engineering: Information Technology for Virtual Enterprises (RIDE-VE 1999), IEEE Proceedings of 9th International Workshop, pp. 4-11

Varma, A., and Agogino, A. (1996). A Machine Learning Approach to Automated Design Classification, Association and Retrieval, *Artificial Intelligence in Design*, John Gero and Fay Sudweeks, (eds.), pp. 429-445.

Virtual View (1996), Technical Manual, Resolution Technology Inc.

Vittich, V., Skobelev, P. (1998). Multi-agent systems for modelling of self-organization and cooperation processes, *Proc. of XIII International Conference on the Application of Artificial Intelligence in Engineering*, Galway, Ireland, pp.91 – 96.

Voit, E., Endogur, A., et.al (1987). *The Design of Aircraft Structures* , Mashinostroenie (Mechanical engineering) Publish. Moscow, Russia, (In Russian).

Wagner, T., Shapiro J., Xuan., P., Lesser., V. (1999). Multi-level conflict in multi-agent systems, AAAI Workshop on Negotiation in Multi-agent systems.

Wallace, G., Sackett, P. (1996). Integrated design for low production volume, large, complex products, *Integrated Manufacturing Systems*, 7/3, 5-16.

Williams, T., Birchall, B., Rishton, J. (1999). Composite design reduces engineering time on Eurofighter parts, *Aerospace engineering and Aerospace Technology*. 71(1), pp.25-28.

Winner, R., Pennell, J., Bertrand, H., and Slusarezuk, M. (1988). The Role of Concurrent Engineering in Weapons System Acquisition, *IDA Report R-338, Institute for Defense Analyses*, Alexandria VA.

Whitney, D., The U.K. Government Program in Engineering Design Research http://web.mit.edu/ctpid/www/Whitney/Europe/hills1.html

Woldridge, M. (2000). Semantic issues in the verification of agent communication language, *Autonomous agents and multi-agent systems*. 3, 9-31.

Volker, W. et. al. (1999) On Conflicts and Negotiation in Multi-User Applications; in: Kent, A.; Williams, J. G. (eds): Encyclopedia of Microcomputers, Marcel Dekker, New Basel, Vol. 23, Suppl. 2, 1999, pp. 63 - 88

Xu, D., and Deng, Y. (2000). Modeling Mobile Agent Systems with High Level Petri Nets. IEEE International

Conference on Systems, Man, and Cybernetics'2000.

Xu, W, Wei, Y, & Fan, Y (2002). *Virtual Enterprise & its Intelligence Management*, Computers & Industrial Engineering, Vol. 42, Issue 2-4, pp.199

Yongtong, H., **Ping, L et al.** (1996), A Multi-agent System for the Support of Concurrent Engineering, IEEE International Conference on Systems, Man and Cybernetics, Vol. 2, pp.959-964

ZEUS Technical Manual (1999), BT Lab, 1999.

APPENDIX A

THE TYPOLOGY OF DISTRIBUTED KNOWLEDGE BASE

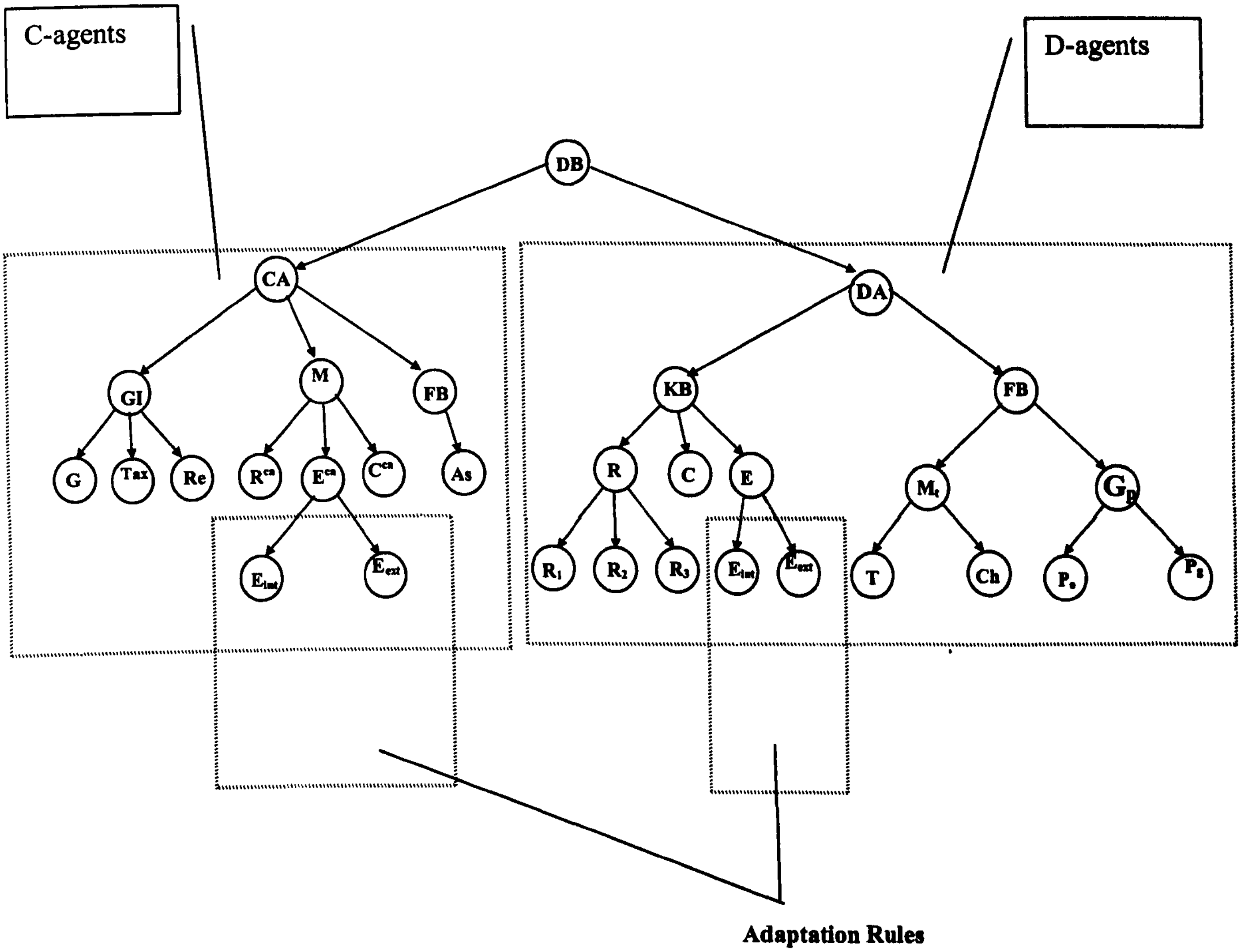


Figure 38: The typology of distributed knowledge base of IDMCS

The description of elements of typology is outlined below:

DB-distributed Knowledge-base
CA –control agent
GI – general meta-knowledge
G –design goals
Re – restrictions
Tax- taxonomy of design mismatches
M – Meta-knowledge base
R - receptors
E -effectors
E – external effectors
E – internal effectors
C -classificators
FB – fact-base of C-agents.
As - assembly sequence
DA – design agent
KB – knoweledge- base of design Agents
$\mathbf{R}_1$, $\mathbf{R}_2$,$\mathbf{R}_3$ - different types of receptors
C – classificators of D-agents
E –effectors of D-agents
$\mathbf{E}_{int}$ – internal effectors of D-agents
$\mathbf{E}_{ext}$ – external effectors of D-agents
FB – fact-base of D-agents
Mc – material characteristics
T – material type
Ch –geometric data
Po – orientation information

APPENDIX B

INTELLIGENT DISTRIBUTED MISMATCH CONTROL.

INDUSTRY QUESTIONNAIRE

Interviewee

Background

____

____

Role

____

____

Experience

____

____

Whom does he interact with?

____

____

Are mismatches easier to resolve between him and his direct colleagues than between others?

____

____

Statistics

When does Mismatch typically occur?

- Geometrical
- Material
- Manufacturing
- Assembly
- Avionics
- Make sure the list is complete

____

____

[Are there other types of mismatches not in this list?
Are these clear and natural categories of mismatches? If not, what types of mismatches would you thinks there are? Are any of these categories too coarse – would you naturally use more precise categories?]

Can you give a rough estimate of the percentage of cases falling into each category?

What is the cost associated with it?
In hours, money

Can you give me an example, ideally an example of a mismatch in each group?

Do mismatches occur within your company/organisation?

Do mismatches occur in between your company and suppliers?

When mismatches occur, which individuals/groups/organisations are on each side?

Mismatches and Suppliers

- What kind of suppliers:
  - in which industry sector
  - between which industry sectors
  - between which countries
- Have you got any idea why?
- Who defines the interfaces between components?
- What measures are taken to avoid mismatches?
  - contrast to in-house procedures

Detection of Mismatches

Who typically detects mismatches?

- computer?
- Manager?
- Designer of one component?
- Designer who uses a mismatched components

When?

- what is typical
- when is the latest
- when should they be detected

Are their procedures in place to detect mismatches?

- Systematic checking?
- Topic in design review meetings
- Informal negotiations

How much cost for delay / rework?

Who gets blamed?

Handling mismatches

What is the procedure in your company for dealing with mismatches?

Is this the same across the company?

Do you record mismatches and analyse them systematically across the company?

Does dealing with mismatches involve different groups of people or different procedures from other design activities?

Negotiation

Whom do you negotiate with?

- Managers?

- Designers who designed part?
- Designers who do rework?

____

____

Does the negotiation involve all concerned parties?

____

____

Do you know your tolerance margins for a change?

____

____

Who has the overview over the product to make balanced changes?

____

____

Are the changes negotiated and solved locally, or does someone with overall responsibility need to deal with the problem?

____

____

Do you have meetings to solve mismatch problems? Do you talk on the telephone, use email, write memos?

____

____

____

____

Do you have meetings or communicate in order to understand the nature of the mismatch problem, or what others think the problem is?

____

____

If there are discussions: How formal are they? Who participates? What is considered (do you try to solve the problem together, or talk about what the problem is, or what)?

____

____

____

What are the outcomes of the discussions? (Any decisions, or progress towards deciding what's possible, or being better informed about the nature of the problem?)

Do you use CAD models, other drawings, sketches or other diagrams to understand the mismatch problem, or to communicate information about it? Do you use them to communicate proposed solutions? Who produces what models/sketches/diagrams, and for what purpose?

Is there any activity recognisable as proposing solutions, evaluating proposed solutions and making counter-proposals? If so, where does it happen, what form do the proposals take, and how do the participants communicate certainty, confidence, degree of commitment etc?

How do you know what can and cannot be changed? How do you know how committed people are to a particular part of the design?

If there are negotiations, who arbitrates negotiations?

Decision making

Who decides what should be changed?

- Does he/she have technical competence?

Note: if negotiations go through purchasing department then this does not need to be the case

Is the decision about what gets altered based on organisational or technical criteria? E.G. Does the supplier also have to make changes? Are the changes made by the person for whom it is easiest?

____

____

Rework

What is the process for making changes due to mismatches?

____

____

Who pays for it?

____

____

What happens to knock-on effects of changes?

____

____

Is there any procedure for identifying knock-on effects? If so, how is it done?

____

____

Are the knock-on effects considered in the negotiations?

____

____

Software

Do you have computer programs to detect mismatches? What role if any do CAD systems play in finding or resolving mismatches?

____

____

What about the knock-on effects?

____

____

How good are they?

____

What would you like a computer program for mismatch detection to be able to do?

Distributed design

What is the effect on the design process or on mismatch handling when organisation is distributed?

Is the process different?

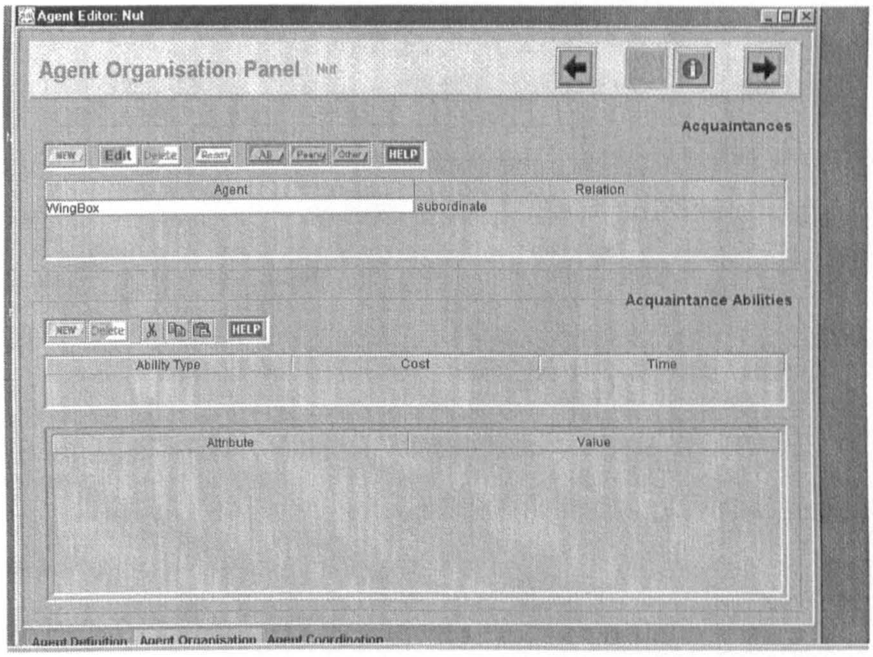
Is it more expensive?

Does it take longer?

Is it more hierarchical?

APPENDIX C

INTELLIGENT DISTRIBUTED MISMATCH CONTROL SYSTEM. AGENT GENERATION STAGE

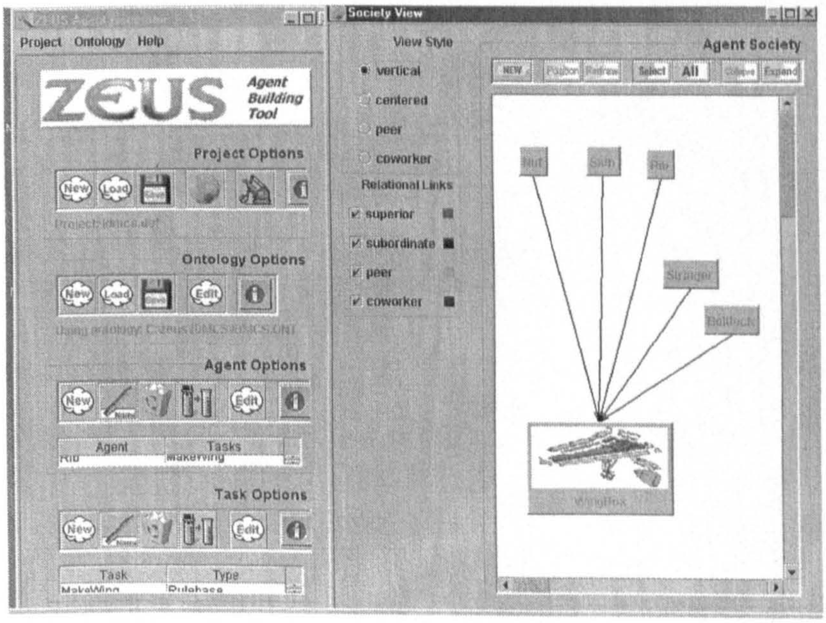


Agent	Relation
WingBox	subordinate

Ability Type	Cost	Time

Attribute	Value

Figure 39: Agent Generator and Agent Society



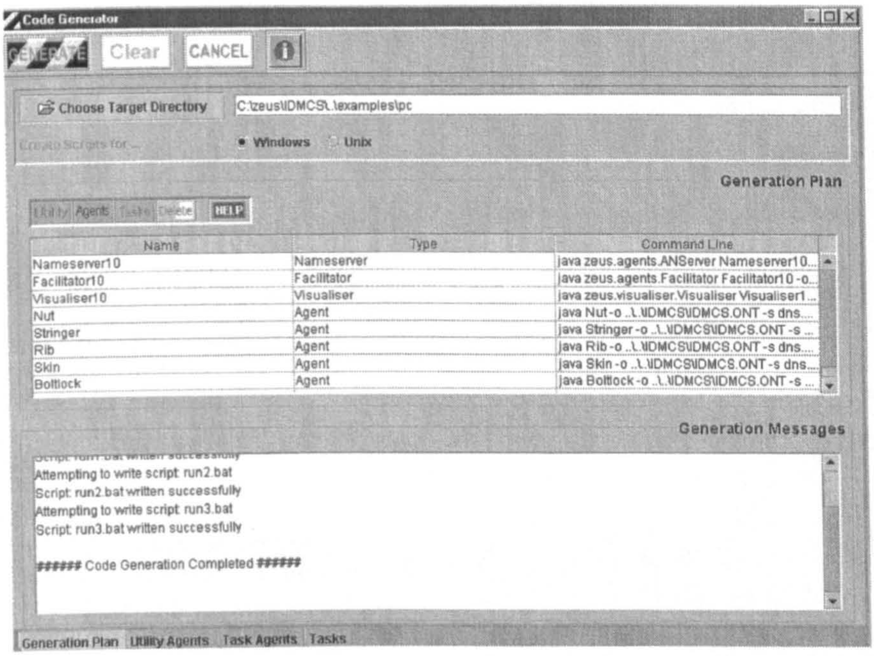
Agent	Tasks
Rib	MakeWing

Task	Type
MakeWing	Rulebase

View Style
● vertical
○ centered
○ peer
○ coworker

Relational Links
☑ superior
☑ subordinate
☑ peer
☑ coworker

Figure 40: ZEUS Agent Generator and Society View



Choose Target Directory: C:\zeus\IDMCS\..\examples\pc

Create Scripts for ... ● Windows ○ Unix

Generation Plan

Name	Type	Command Line
Nameserver10	Nameserver	java zeus.agents.ANServer Nameserver10...
Facilitator10	Facilitator	java zeus.agents.Facilitator Facilitator10 -o...
Visualiser10	Visualiser	java zeus.visualiser.Visualiser Visualiser1 ...
Nut	Agent	java Nut -o ..\..\IDMCS\IDMCS.ONT -s dns....
Stringer	Agent	java Stringer -o ..\..\IDMCS\IDMCS.ONT -s ...
Rib	Agent	java Rib -o ..\..\IDMCS\IDMCS.ONT -s dns....
Skin	Agent	java Skin -o ..\..\IDMCS\IDMCS.ONT -s dns....
Bolttock	Agent	java Bolttock -o ..\..\IDMCS\IDMCS.ONT -s ...

Generation Messages

```
Script run1.bat written successfully
Attempting to write script run2.bat
Script run2.bat written successfully
Attempting to write script run3.bat
Script run3.bat written successfully

###### Code Generation Completed ######
```

Figure 41: Code Generator

IDMCS Code Example

```
/*
        This software was produced as a part of research
        activities. It is not intended to be used as commercial
        or industrial software by any organisation. Except as
        explicitly stated, no guarantees are given as to its
        reliability or trustworthiness if used for purposes other
        than those for which it was originally intended.

        (c) British Telecommunications plc 1999.
*/

/*
This code was automatically generated by ZeusAgentGenerator version 1.01
                DO NOT MODIFY!!
*/

import java.util.*;
import java.io.*;
import zeus.util.*;
import zeus.concepts.*;
import zeus.actors.*;
import zeus.agents.*;

public class WingBox {
  protected static void version() {
    System.err.println("ZeusAgent - WingBox version: 1.01");
    System.exit(0);
  }

  protected static void usage() {
    System.err.println("Usage: java WingBox -s <dns_file> -o <ontology_file> [-gui ViewerProg] [-e
<ExternalProg>] [-r ResourceProg] [-debug] [-h] [-v]");
    System.exit(0);
  }

  public static void main(String[] arg) {
    ZeusAgent agent;
    String external = null;
    String dns_file = null;
    String resource = null;
    String gui = null;
    String ontology_file = null;
    Vector nameservers = null;
    Bindings b = new Bindings("WingBox");
    FileInputStream stream = null;


    for( int j = 0; j < arg.length; j++ ) {
      if ( arg[j].equals("-s") &&  ++j < arg.length )
        dns_file = arg[j];
      else if ( arg[j].equals("-e") &&  ++j < arg.length )
        external = arg[j];
      else if ( arg[j].equals("-r") &&  ++j < arg.length )
        resource = arg[j];
      else if ( arg[j].equals("-o") &&  ++j < arg.length )
        ontology_file = arg[j];
      else if ( arg[j].equals("-gui") &&  ++j < arg.length )
        gui = arg[j];
      else if ( arg[j].equals("-debug") ) {
        Core.debug = true;
        Core.setDebuggerOutputFile("WingBox.log");
      }
      else if ( arg[j].equals("-v") )
        version();
      else if ( arg[j].equals("-h") )
        usage();
      else
        usage();
    }

    if ( ontology_file == null ) {
      System.err.println("Ontology Database file must be specified with -o option");
      usage();
    }
    if ( dns_file == null ) {
      System.err.println("Domain nameserver file must be specified with -s option");
      usage();
    }

    try {
      nameservers = ZeusParser.addressList(new FileInputStream(dns_file));
      if ( nameservers == null || nameservers.isEmpty() )
        throw new IOException();

      agent = new ZeusAgent("WingBox",ontology_file,nameservers,1,20,false,true);

      AgentContext context = agent.getAgentContext();
      OntologyDb db = context.OntologyDb();


/*
      Initialising Extensions
*/
      Class c;

      if ( resource != null ) {
        c = Class.forName(resource);
        ExternalDb oracle = (ExternalDb) c.newInstance();
        context.set(oracle);
        oracle.set(context);
      }
      if ( gui != null ) {
        c = Class.forName(gui);
        ZeusAgentUI ui = (ZeusAgentUI)c.newInstance();
        context.set(ui);
        ui.set(context);
      }

/*
      Initialising ProtocolDb
*/
      ProtocolInfo info;

/*
      Initialising TaskDb
*/
      AbstractTask t;
      stream = new FileInputStream("MakeWing.clp");
      t = ZeusParser.reteKB(db,stream);
      stream.close();
      if ( t.resolve(b) )
        agent.addTask(t);
```

```
/*
      Initialising OrganisationalDb
*/
      AbilityDbItem item;
/*
      Initialising ResourceDb
*/
      Fact f1;


/*
      Initialising External User Program
*/

      if ( external != null ) {
        c = Class.forName(external);
        ZeusExternal user_prog = (ZeusExternal) c.newInstance();
        context.set(user_prog);
        user_prog.exec(context);
      }

/*
      Activating Rete Engine
*/
      context.ReteEngine().run();
    }
    catch(Exception e) {
      e.printStackTrace();
      System.exit(0);

    }

  }

}
```

APPENDIX D

INDUSTRIAL COLLABORATORS INFORMATION, LETTERS OF SUPPORT AND IMPLEMENTATION RESULTS

Aviation Euro-Russian Consortium

Aviation Euro-Russian Consortium

Petrovka Str. 24

Moscow, Russia

Tel: +007 (095) 311-07-41

Major Russian aviation enterprises in 1997 formed their own consortium to negotiate deals for major chunks of the A380 production program.

The consortium, known as the Aviation Euro-Russian Consortium unites the Economic Development and Trade Ministry, the Aviastar, Hydromash and Tupolev design and production enterprises, as well as the NIAT and TsAGI research institutes from Russian side and BAE Systems and AIRBUS from Western side.

Tupolev Air Scientific And Technical Complex

Tupolev Aviation Complex JSCo
17 Tupolev Embankments,
111250 Moscow, Russia
Tel.: + 7 095 267 2508,
Fax: + 7 095 261 0868, 261 7141

The Russian Aviation Consortium [Rosaviaconsortium RAC] financial-industrial group (FIG) was set up in May 1995. The structure comprised over 14 companies and organizations, including the Tupolev Aviation Scientific-Technical Complex, the Ulyanovsk Aviastar Aviation-Industrial Complex, the Perm Motors joint stock company, the KAPO named after Gorbunov, the Vnukovo Air Lines air company, the Donavia joint stock company, the Universal Scientific-Industrial Center and the Promstroybank of Russia. ANTK worked with Boeing and NASA to design a supersonic passenger jet on the basis of Tu-144LL and with "Airbus Industrie" to develop a high capacity European aircraft - A380.

Tupolev Air Scientific And Technical Complex and Ulyanovsk "Aviastar" air-factory were consolidated in a uniform structure according to the Government of Russia decree of 30 July 1999, on the basis of the two enterprises there will be formed OAO "Tupolev", to which permanent and other assets of the enterprises were transmitted.

The Tupolev Aviation Scientific-Industrial Complex (ANTK) is ready to launch full-scale development program of the Tu-324 aircraft. The Tu-324's mock-up and initial design have been already made.

This turbojet aircraft is supposed to have a broad sphere of application in Russia. The aeroplane's tourist-carrying 50-seat version will have a flight range of 2,500 kilometres; a two-class 44-seat version - 3,000 kilometres, an administrative 8-9-seat version - over 7,000 kilometres with no refuelling. According to Dmitriyev, the Tu-324's jet flight characteristics make this aeroplane superior to the Russian-Ukrainian and Russian-Uzbeki An-140 and Il-114 turbojet aircraft, providing for its future leading position in air transportation market.

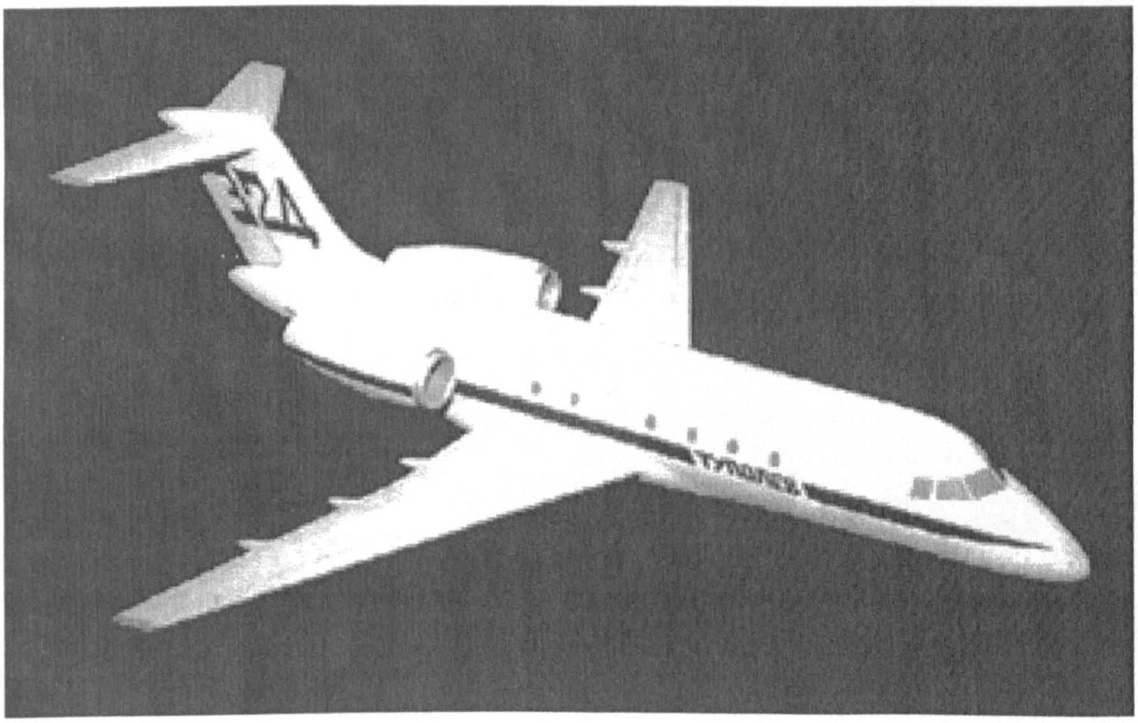


Figure 42: TU-324 aircraft

The IDMC-approach was used in development of TU-342 aeroplane.

Aviastar (Ulyanovsk Aviation Industrial Complex)

AO "Aviastar"
Ulyanovsk Aviation Production Complex
Ulyanovskiy aviatsionnyy promyshlennyy kompleks imeni D. F. Ustinova

Prospekt Sozidateley, 9; Prospekt Antonova, 1
432062 Ulyanovsk, Russia
Tel: (8422) 20-72-26
Fax: (8422) 20-95-61

The Ulyanovsk Aviation Production Complex - Aviastar a member of the ANTK Tupolev production group, is a big, new and well equipped aircraft assembler. That is the largest aviation production facility in the world and is the newest one in Russia and was originally intended to have airframe, avionics, and engine manufacturing facilities all in one complex.

Aircraft of the TU-series are produced at Aviastar, KiGAZ, Aviacor, KAPO Gorbunova, Takom-Avia and Arnaks aircraft factories. It produces the An-124 long-range heavy transport aircraft (comparable with the U.S. C-5) and the 200-seat Tu-204 medium-range airliner (comparable with the Boeing 757).

A major investment program has enabled Aviastar to acquire advanced equipment including a completely automatic GEMCOR wing manufacturing plant, which is one of the most, advanced in the industry. The automated electrostatic paint facility is capable of handling the largest aircraft currently manufactured anywhere. A full range of laboratory, testing and analytical equipment provides a most comprehensive capability for any aircraft manufacture. Extensive use is made of the most modern computer-driven design capability and computer-controlled manufacturing processes using dedicated software designed exclusively for this purpose.

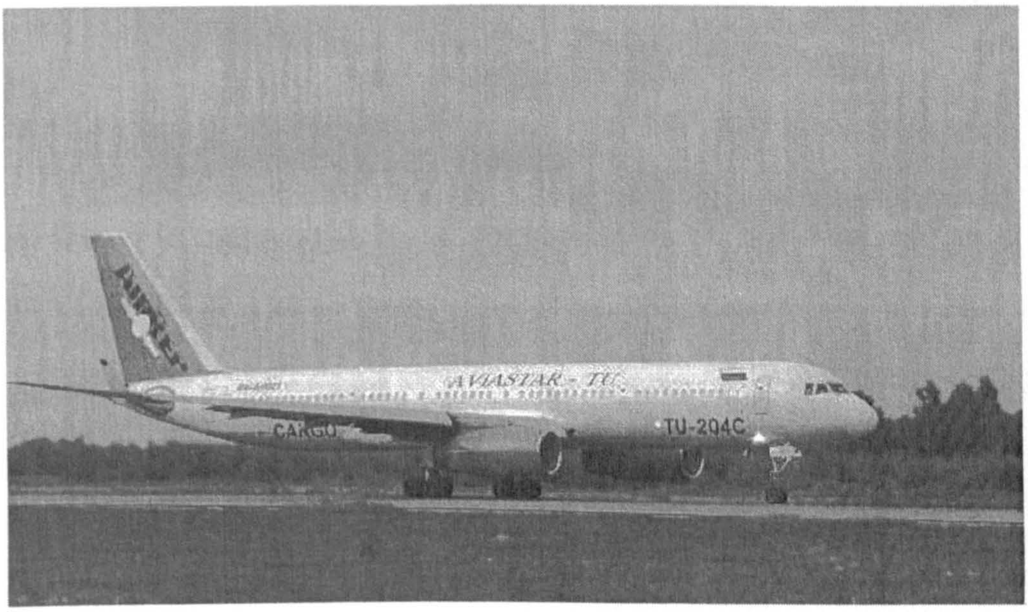


Figure 43: TU-204C Aircraft (Photo © Lars Walhstrom)

“AERC”
public limited company
“AVIATION EURO-RUSSIAN CONSORTIUM”

To:		***From***	
Company:	**Enterprise Integration** **Cranfield University**	***Company:***	**AERC**
Name:	**Dr. Victor Taratoukhine**	***Name:***	**Libkind E.**
Copy:	Lecturer in Decision Engineering		**Executive Director**
Department:		***Department:***	
Telephone:	**+44 1234 754 072**	***Telephone:***	**(+7 095) 311-07-41**
Fax:	**+44 1234 750 852**	***Fax:***	**(+7 095) 311-07-41**
Your reference		***Our reference***	**№** КН-02Б/402
Pages:	**2**	***Date:***	09.11.01

Dear Sirs,

Re: Viktor Vladimirovich Taratoukhine, Research Project
“A Multi-agent approach for design consistency checking”

I am writing to confirm our support to the research conducted by Victor Vladimirovich Taratoukhine.

Aviation Euro-Russian Consortium is main coordinated structure between European and Russian Aerospace Industry. Our participants are AIRBUS, BAE Systems, Tupolev Corporation, Ulyanovsk Aerospace Industrial Complex (AVIASTAR), National Institute of Aviation Technologies and TSAGI Institute.

Victor Taratoukhine’s research addresses the theoretical and practical needs to develop a new methodology for collaborative and distributed design in aerospace industry. A multi-agent approach and other Artificial Intelligence methods are used to produce this modern intelligent distributed CAD software. Victor Taratoukhine’s research is a contribution to more than one important industrial applications.

We would be very interested in using the research carried by Victor Taratoukhine to create a distributed design environment for detecting design inconsistencies in Consortium-based projects.

I am fully support this important research.

Yours sincerely,

E. Libkind

Director executive

Aviation Euro-Russian Consortium

Public Stock Company **TUPOLEV**
Building 1, 23, Bakhroushin Street, Moscow, 113054, Russia
Phone: (095) 238-34-13
Fax: (095) 238-68-41
261-71-41
261-08-68

CERTIFIED
by President of TUPOLEV Corporation
A.P.Poliakov
(signature)
(stamp)

CERTIFICATE
relating to the implementation of the results of doctoral work of Victor Vladimirovich Taratoukhine

The Commission: Chairman : V.I. Solozubov, General Director of Central Design Centre.
Members of the Commission: S.A. Bogatikov, Deputy Director of Central Design Centre;
A.S. Slobodchikov, Head of Computer Technology Centre

This document certifies that the results of the research into methodology, methods, models and tools for mismatch control in distributed design complex technical systems were implemented in designs of the TUPOLEV Corporation as part of the development of computer-based models for different elements in the construction of the TU-324 aircraft. The implemented results are as follows:

1. The methodological principles, the organisation of engineering design mismatch control in a distributed design environment;
2. The model of organisation of taxonomy of design mismatches, the principles of organisation of the distributed knowledge-base;
3. The models of mismatch detection based on a multi-agent approach;
4. The general principles of knowledge extraction, the definition of taxonomy of design mismatches. The development of taxonomy design mismatches for aerospace design/assembly/manufacturing;
5. The methods of development of multi-agent CAD systems with intelligent control possibilities;
6. Intelligent Distributed Mismatch Control System;
7. The experimental data obtained from distributed design, statistics relating to the detection of mismatches in the project.

Using these results for engineering design will help to increase the quality and effectiveness of design and to reduce the cost of the engineering design process.

Chairman of Commission	(signature)	V.I. Solozubov
Members of Commission:	(signature)	S.A. Bogatikov
	(signature)	A.S. Slobodchikov

I CERTIFY THIS TO BE AN ACCURATE TRANSLATION OF THE CERTIFICATE.

THE TRANSLATION CONSISTS OF ONE PAGE OF RUSSIAN TRANSLATED INTO ONE PAGE OF ENGLISH.

J. GLADKOW,
EUROPEAN LANGUAGES ~~CENTRE/SCHOOL OF MANAGEMENT,~~
CRANFIELD UNIVERSITY, UK

ОТКРЫТОЕ АКЦИОНЕРНОЕ ОБЩЕСТВО
ТУПОЛЕВ

Российская Федерация, 113054, Москва,
ул. Бахрушина, д. 23, стр.1
тел. 238-34-13, факс 238-68-41
261-71-41
261-08-68

PUBLIC STOCK COMPANY
TUPOLEV

Building 1, 23, Bakhroushin street
Moscow, 113054, Russia
phone: (095)238-34-13, fax: (095) 238-68-41
261-71-41
261-08-68

Утверждаю

Президент ОАО «Туполев»

_____________ [illegible] Поляков

АКТ
о внедрении результатов докторской диссертационной работы Таратухина Виктора Владимировича

Комиссия в составе: председатель Солозобов В.И. – Директор ЦКБ
члены комиссии Богатиков С.А. – Зам. Директора ЦКБ;
Слободчиков А.С. – Начальник Центра компьютерных технологий.

Составили настоящий акт о том, что результаты диссертационной работы «Методология, методы, модели и средства контроля ошибок при распределенном проектировании сложных технических систем» представленной на соискание ученой степени, использованы в проектно-конструкторской деятельности на ОАО «Туполев» при разработке электронных моделей различных элементов самолета ТУ-324 включая точные модели конструктивных элементов.

1. Методологических принципов организации контроля ошибок проектирования при распределенной проектно-конструкторской деятельности.
2. Модели построения таксономий ошибок проектирования, принципов организации распределенной базы знаний.
3. Моделей обнаружения ошибок проектирования на основе многоагентного подхода.
4. Принципов организации процедур извлечения знаний, построения таксономии ошибок проектирования в области разработки сложных систем в самолетостроении.
5. Методик построения многоагентных САПР со встроенным механизмом обнаружения конфликтов в проекте.
6. Интеллектуальной распределенной системы контроля ошибок проектирования.
7. Экспериментальных данных по результатам распределенной экспертизы проекта, статистики по обнаружению и устранению конфликтов в проекте.

Использование указанных результатов позволяет повысить качество проектирования, существенно сократить количество ошибок при проектировании на всех этапах работ, что сокращает затраты на проведение ~~опытно-~~конструкторских работ и ускоряет работы.

Председатель комиссии В.И. Солозобов

Члены комиссии С.А. Богатиков

А.С. Слободчиков

Join Stock Company
AVIASTAR –SP
Russian Federation
432072, Ulyanovsk,
Antonov Str., 1
Tel: 8422-29-11-20
Fax: 8422-20-25-75

CERTIFIED
by Executive Manager
AVIASTAR-SP
V.V. Maltsev
(signature)
(stamp)

Date: 28.05.2001
Reg. No. 110/2764

CERTIFICATE

relating to the implementation the results of doctoral work of
Victor Vladimirovich Taratoukhine

The Commission: Chairman: Mr V.A. Donetsky, Chief Designer.
Members of the Commission: Mr Y.M. Avetikov, Deputy Designer, responsible for General Engineering Design and CAD systems; Mr A.A. Chernov, Head of CAD Centre.

This document certifies that the results of the research into methodology, methods, models and tools for mismatch control in the distributed design of complex technical systems were implemented in the Design Centre, Department 110 of the Chief Designer in the development of a geometric model of the fuselage of the TU-204C aircraft. The implemented results are:

1. The methodological principles, the organisation of engineering design mismatch control in a distributed design environment;
2. The model of organisation of taxonomy of design mismatches, the principles of organisation of the distributed knowledge-base;
3. The models of mismatch detection based on a multi-agent approach;
4. The general principles of knowledge extraction, the definition of taxonomy of design mismatches. The development of taxonomy design mismatches for aerospace design/assembly/manufacturing;
5. The methods of development of multi-agent CAD systems with intelligent control possibilities
6. Intelligent Distributed Mismatch Control System;
7. The experimental data obtained from distributed design, statistics relating to the detection of mismatches in the project.

Use of these results for engineering design will help to increase the quality and effectiveness of design to reduce the cost of the engineering design process.

Chairman of Commission	(signature)	V.A. Donetsky
Members of Commission:	(signature)	Y.M. Avetikov
	(signature)	A.A. Chernov

I CERTIFY THIS TO BE AN ACCURATE TRANSLATION OF THE CERTIFICATE.

THE TRANSLATION CONSISTS OF ONE PAGE OF RUSSIAN TRANSLATED INTO ONE PAGE OF ENGLISH.

J. GLADKOW
EUROPEAN LANGUAGES CENTRE, SCHOOL OF MANAGEMENT,
CRANFIELD UNIVERSITY, UK

ЗАКРЫТОЕ
АКЦИОНЕРНОЕ
ОБЩЕСТВО

«АВИАСТАР-СП»

РОССИЙСКАЯ ФЕДЕРАЦИЯ
432072, г. Ульяновск,
Антонова 1,
Тел./факс: (8422)-29-11-20
Факс: (8422)-20-25-75

28.05.2001 № 110/2764

на № ____________ от ______________

УТВЕРЖДАЮ
и.о. Генерального директора
ЗАО «Авиастар-СП»
______________ [illegible] Мальцев [illegible]енко О.Ю.
«28» ________ 2001 г.

АКТ

о внедрении результатов докторской диссертационной работы Таратухина Виктора Владимировича

Комиссия в составе: председатель Донецкий В.А. – Главный конструктор
члены комиссии Аветиков Ю.М. - Зам. Главного конструктора по ОКР и САПР;
Чернов А.А. – Начальник КБ САПР.

Составили настоящий акт о том, что результаты диссертационной работы «Методология, методы, модели, и средства контроля ошибок при распределенном проектировании сложных технических систем» представленной на соискание ученой степени, использованы в проектно-конструкторской деятельности КБ САПР отд. 110 УГК при разработке геометрической модели панели фюзеляжа Ту-204С в виде:

1. Методологических принципов организации контроля ошибок проектирования при распределенной проектно-конструкторской деятельности.
2. Модели построения таксономий ошибок проектирования, принципов организации распределенной базы знаний.
3. Моделей обнаружения ошибок проектирования на основе многоагентного подхода.
4. Принципов организации процедур извлечения знаний, построения таксономии ошибок проектирования в области разработки сложных систем в самолетостроении.
5. Методик построения многоагентных САПР со встроенным механизмом обнаружения конфликтов в проекте.
6. Интеллектуальной распределенной системы контроля ошибок проектирования.
7. Экспериментальных данных по результатом распределенной экспертизы проекта, статистики по обнаружению и устранению конфликтов в проекте.

Использование указанных результатов позволяет повысить качество проектирования и эффективность конструкторских работ, сократить затраты на проведение опытно-конструкторских работ.

Председатель комиссии В.А. Донецкий

Члены комиссии Ю.М. Аветиков

А.А. Чернов

APPENDIX E

VICTOR TARATOUKHINE. SELECTED PUBLICATIONS

Taratoukhine, V., (2001a). An Agent-based methodology for consistency checking in engineering design, International Congress, AI in 21st Century, Gelendgik, Black Cost, Russia, 2001

Taratoukhine, V., et. al. (2001). Computer supported collaborative design: mismatch control, methodology and application, 8th International Workshop of the European Group for Structural Engineering Applications of Artificial Intelligence (EG-SEA-AI), UK, July, 2001

Taratoukhine, V., (2001b). Using distributed artificial intelligence for the support of the design process, 4th International Conference on Soft Computing and Measurements, SCM'2001, June 2001

Taratoukhine, V., Bechkoum, K., Stacey, M. (2001a). Virtual Design Team Management. Consistency checking process, New Models of business: managerial aspects and enabling technology, St. Petersburg State University and

University of California, Berkeley, June, 2001

Taratoukhine, V, Bechkoum, K., Stacey, M (2001b). Intelligent Distributed Mismatch Control for European Aerospace Industry, CAD/CAM/PDM - '2001, The International Conference and Exhibition, Institute of Control Sciences, Russian Academy of Sciences, Moscow, April, 2001

Taratoukhine, V., Bechkoum K., (2000a). An agent-based paradigm to support virtual design mock-ups, 14th European Conference on Artificial Intelligence, Workshop on Applied Semiotics, Berlin, Germany, 2000

Taratoukhine, V., Bechkoum, K., (2000b). Intelligent Distributed Mismatch Control, CSCMP-2000 Complex Systems: Control and Modeling Problems, 19-24 June 2000 Samara, Russia

Taratoukhine, V., Bechkoum, K., (2000c).Intelligent Distributed Mismatch Control: Model and Application, 3rd International Conference on Soft Computing and Measurements, SCM'2000, June 27-30, St. Petersburg, Russia

Taratoukhine, V., (2000). Intelligent Detection of Mismatches in Distributed Design, Intelligent CAD International Conference, Journal of Taganrog Radiotechnical University, Vol 1, January 2000, Taganrog, Russia

Bechkoum, K., Taratoukhine, V., (1999a). Software Agents to Support Geometric Modelling., Expert Update (BCS SGES), June 1999.

Taratoukhine, V., Bechkoum, K., (1999a). Towards a Consistent Distributed Design: A Multi-Agent Approach, Proc Information Visualisation '99, London, IEEE Press, 1999.

Taratoukhine, V., Bechkoum, K., (1999b). Mismatch Detection in Distributed Computer Aided Design", Proc International Conf. On Computer Graphics & Vision, ACM SIGRAPH and Lomonosov Moscow University, Moscow, 26 August-1st Sep. 1999.

Taratoukhine, V., Bechkoum, K., (1999c).Mismatch Detection in Distributed Computer Aided Design", International Journal Computer Graphics and Geometry, Vol. 2, No. 3. St. Petersburg, Russia (reprint from Proc. International Conf. On

Computer Graphics & Vision, ACM SIGRAPH)

Bechkoum, K., Taratoukhine, V., (1999b). A Framework for Mismatch Control in a Distributed Design Environment., Proc. Advances in Concurrent Engineering, Bath, 1-3 September 1999.

Taratoukhine, V., Bechkoum, K. (1999d). Semiotic Multi-agent Framework for Intelligent Mismatch Detection in Distributed Design, Proc. 4th Int'l Workshop on Applied Semiotics and Semiotics Control ASC'99, Zelenograd, Russia, 53-54 pp.

Taratoukhine, V., Bechkoum, K. (1999e). A Multi-Agent Approach for Design Consistency Checking, Proc. International Conference Human Computer Interaction - 99, Ulyanovsk, Russia, 20-22 September 1999. 21-22 pp.