

# Audit, Control and Monitoring Design Patterns (ACMDP) for Autonomous Robust Systems (ARS)

Antoine Trad<sup>1</sup> and Catherine Trad<sup>2</sup>

- 1) TradSoft, 1007 Lausanne, Switzerland, [antoine.tradsoft@bluewin.ch](mailto:antoine.tradsoft@bluewin.ch); [www.tradsoft.info](http://www.tradsoft.info).
- 2) TradSoft, 1007 Lausanne, Switzerland, [catherine.tradsoft@bluewin.ch](mailto:catherine.tradsoft@bluewin.ch); [www.tradsoft.info](http://www.tradsoft.info).

---

**Abstract:** This paper proposes the Audit, Control and Monitoring Design Patterns (ACMDP) for building Autonomous and Robust Systems (ARS) such as Mobile Robot Systems (MRS). These patterns are also applicable to other Mission Critical and Complex Systems (MCCS). This paper presents a proposal which will help ARS project managers and engineers design, build and estimate the probability that an ARS will succeed or fail. Furthermore, this proposal offers the possibility to detect in advance potential ARS problems with the help of audit, monitoring and controlling components, adjust the project management pathways, and define the problem sources as well as their possible solutions, in order to deliver an ARS or an MRS.

**Keywords:** ARS, ARS or an MRS, Design Patterns, Packages, Components, Maintenance, Audit, Control and Monitor.

---

## 1. Introduction

This paper proposes an ACMDP for building, auditing, controlling and monitoring ARS or MRS from an Information and Communication Technology (ICT) point of view (Trad, A., Kalpic, D. & Fertalj, K., 2002).

These “cross-platform” patterns are also applicable to other complex autonomous systems that need high availability and robustness levels. These systems are autonomous in their logic, processing and energy supply (Krishnamurthy, B. & Rexford, J., 2001).

The paper results in an ACMDP “requirements proposal” which should help ARS Engineers (ARSE or MRS Engineers) to estimate the probability that the construction and the maintenance of an ARS (or an MRS) will succeed or fail (Trad, A. & Dessimoz, J.-D., 2004).

Furthermore, this proposal offers a possibility to forecast possible ARS (and MRS) problems, launch corrective actions as proactive scripts (panic procedures) (Drake C. & Brown, k., 1995), adjust project management pathways and define problem sources as well as their feasible solutions. This possible solution(s) will be handed over to system

developers for correction (Watters, P. & Veeraraghawa 2000).

Although we have many languages tools, standards (Papurt, D., 1995) and methods (Burkhardt, R., 1997) for designing and implementing most of ARS and MRS software components, we still do not have «off the shelf specialized» applicable interactive tools, system design methods and Design Patterns (DP) in the areas of building and controlling (Jones, 1996) complex ARS and MRS, or other MCCS as a whole.

This «requirements» proposal is primarily intended to be of interest to higher-level technical personnel, professors, auditors and senior managers in charge of critical ARS and MRS.

### 1.1 Paper's structure

The paper is composed of the following sixteen sections:

*The Automated Manufacturing Environments Method (AMEM); AMEM's Team Design (AMEMTD); The Generic and Infrastructural Design Patterns (GIDP); The Specialized Design Patterns (SPECDP); The Communication Design Patterns (COMMDP); The Data Storage Design*

*Pattern (DSDP); The Vision Recognition Design Pattern (VRDP); The Decision Making Design Pattern (DMDP); The Human Robot Interaction Design Pattern (HRIDP); The FES Design Pattern (FESDP); The Audit, Control and Monitoring Design Patterns (ACMDP); The Underlying Architecture; ACMDPs System's Architecture; ACMP Maintenance; An MRS Implementation; Conclusion and References.*

## 1.2 Acknowledgments

The author assumes that the reader is familiar with basic ARS (or MRS); ICT infrastructure; object oriented; and development methodologies (and some design patterns basics) techniques.

We would like to acknowledge the work of the referenced authors of the publications that were used to compile this article.

A special thanks to the "ARS Journal" that enabled us to publish this article.

Any remaining mistakes or errors in the text are the responsibility of the authors.

## 2. The AMEM

A "Method" is a procedure for attaining an objective in accordance with a particular theory. Whereas a "Methodology" is the branch of knowledge that deals with a method and its application in a particular field (Brown L., 1997).

Of course this procedure can be related to the "simple" choice of material for a future screw driver product but it can also concern a much less simple automation procedure for car manufacturing process; that is why the AMEM must deliver a "generic" notation and a set of artifacts to describe various types of procedures (Nance R. & Arthur J., 1988).

### 2.1 ISRQCC the AMEM predecessor

The Information System Risk and Quality Check Coefficient (ISRQCC) is a step-by-step methodology that can help you to plan and implement an iterative process of Information Systems (IS) auditing within any stage of the development, implementation and maintenance of systems from various problem domains.

Most organizations will be challenged to use their IS audit and control results in order to change their business operations, re-engineer their IS, or to re-schedule various tasks of project management plans, which could result in automating tasks that might have been performed manually in the past. Various solutions as a result of IS auditing process could be offered: from implementing new development paradigms and emerging technologies, to solutions that are based on legacy systems as a better balance between costs, benefits and risk.

However, in all cases the ISRQCC will enable you to have a clearer idea of what your businesses' and IT needs are, and will give you a clear list of all benefits of using different technologies for developing an IT infrastructure that supports its strategic business objectives.

This might be of significance when knowing that only 26% of software projects succeed when developing IS. Hence the IS audit activities are becoming a very common intervention to 'save' them from failure (Trad, A., 1995).

The ISRQCC audit and control method is intended to be of interest to non-technical auditors, audit managers, audit committee members, senior managers in charge of critical computing systems such as robot systems, executives, board members, and even seasoned IS auditors (Trad, A., Kalpic, D. & Trad, C., 2002). This audit method results in a **heuristic model** which will help information system auditors to estimate the probability that an IS will succeed or fail (Dayton, D., 1999). Furthermore the ISRQCC offers the possibility to forecast IS problems, adjust the project management pathways and define the problem's source(s) as well as its possible solution(s). Although we have many tools and standards (ISACA-S, 1998) for designing and implementing most of the IS components, until today we still do not have applicable interactive tools, methods or theories in the areas of estimating and auditing of risks, costs, feasibility, viability and hence quality of complex IS. The ISRQCC concentrates on the project **feasibility** and on finding the set of possible solutions for the current and future problems. This paper presents the use of the of ISRQCC audit method to audit the system, and help the project manager and designer in establishing competitive maintenance and stabilization procedures. The ISRQCC was used also to define future evolutionary steps; and to avoid the blind and risky method of "let's re-develop" the whole system. In contrary, it reused as much as possible of the existing system (Trad A. & Kalpic, D., 1999).

### 2.2 The AMEM proposal

Increasingly competitive "Automated (Autonomous) Manufacturing Environments" (AME) are the main driving forces for research and improvement in the development of avanguard flexible and efficient methodologies based on IS and specialized in the control and monitoring of ARS and MRS (Trad, A., Perrenoud, A., Gauthey, P-F. & Dessimoz, J.-D., Marcuard, J.-D. & Riedi, M., 2004).

The proposed methodology uses various components which all promote an iterative modeling development process of AMEs, based on a high level of reusability of existing components.

Similar to existing methodologies that are applied in these application domains (Lewis, R., 2004).

The success of such methodologies and the IS development, implementation and maintenance processes strongly influence the way AME processes are managed and carried out; this consequently forces AMEs to evolve and improve. Many *factors* affect the AME business evolution and the corresponding IS (re)engineering processes, these *factors* are based on the organization's production infrastructure.

In this paper the authors propose a methodology for planning and implementing production infrastructures based on DPs specialized in building ARS and MRS; in order to improve AME development process and to fully support company strategic and business needs.

From now on, the "AME Methodology" will be identified as AMEM.

The proposed methodology uses different components (as set of DPs) which promote an iterative development process of AMEs, based on a high level of reusability of existing components.

Components (such as the COMMDP, please do refer to section -6- for more information) of this methodology have been successfully applied and results are implemented in various projects.

The process of gathering and analyzing an application's requirements, and incorporating them into a design procedure, is a complex one and the industry currently supports many methodologies such as the Unified Modeling Language (UML) that define formal procedures for the improvement of AME integration. Regardless of the methodology that is used to perform the analysis and design, UML can be used to express the results (Thramboulidis, K., 2004).

By using XMI (XML Metadata Interchange, another OMG<sup>1</sup> standard) the UML model (Magnus H.E. & Penker M., 1998) can be transferred from one tool into a repository, or into another tool for refinement. These are the benefits of standardization (Selic, B. & Rumbaugh, J., 1999).!

The success of such methodologies and the IS development, implementation and maintenance processes strongly influence the way AME processes are managed and carried out; this consequently forces AMEs to evolve.

AMEs consist in Industrial-grade Personal Computers (IPC), ARSs, PLCs and/or specialized systems. Many *factors* affect AME evolution and the corresponding IS engineering processes, and many of these *factors* are based on the production infrastructure of the organization.

In this section the authors propose a methodology for planning and implementing production infrastructures for ARSs and of course MRs; in order to improve the AME's development process

that fully supports the company's strategic and business needs.

The AME Engineers (AMEE) capture the users' requirements" using standard UML Use Case diagrams (as shown in "Fig. 1.), and Rational Rose modeling tool can be used for that ([www.rational.com](http://www.rational.com), 2004).

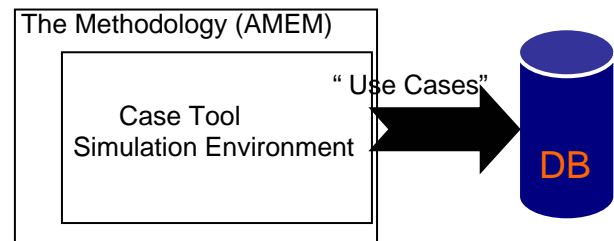


Fig. 1. Presents the AMEM main components.

### 3. AMEM's Team Design (AMEMTD)

A very important factor for AMEM's implementation is the success of the team.

Regardless of the technology, the *human factor* of the team dynamics plays the major role in the success of the ARS (or MRS) design and development team.

This *factor* influences the success an ARS team in all the design, development and maintenance processes.

It establishes a starting point for modelling the effect these *factors* have on team success (Meier, R., 1997).

The author developed the "Team Quality Check Coefficient" (TDQCC) to help project managers and designers in diagnosing some of the problems endemic to ARS (or ICT in general) teams.

It can likewise be used to audit and thus to improve the quality of ARS development and maintenance teams.

AMEMTD (previously known as the TDQCC) stands in contrast to the typical non-empirical ad hoc policy for team building and evaluation (Trad, A. & Kalpic, D., 1998).

### 4. The Generic and Infrastructural Design Patterns (GIDP)

An ARS generic and infrastructural design pattern systematically localizes, names, labels, motivates, and explains a general design that addresses a recurring design problem in the construction of ACMDPs basic design patterns. In general, these design patterns are pure ICT DPs that solve design problems, such as the implementation of the Model View Control (MVC) model

It describes the requirement, domain problem, the solution and when to apply the GIDPs.

<sup>1</sup> Object Management Group (OMG)

The GIDP is a general arrangement of objects and classes that solve basic ACMDP problem(s). The GIDP is customized and implemented to build the ACMDP in a particular context. The GIDP is strongly influenced by «Gang of Four» (GoF) design patterns (Gamma, E., Helm, R., Johnson, R. & Vlissides, J., 1995).

The GIDP, are elements of the ACMDP reusable basic components. This article is the first version of a series of articles illustrating the ACMDP (known previously as the ISRQCC).

In this implementation the author implemented the Abstract Factory, MVC, Factory Method, Builder patterns and the Singleton pattern.

## 5. The Specialized ARS DP (SPECDP)

Mature engineering disciplines have handbooks and methods that describe successful solutions to known problems. For instance, automobile designers do not design cars using the laws of physics. Instead, they reuse standard designs with successful track records. The extra few percent of performance available by starting from scratch typically isn't worth the cost.

In the previous section we described the GIDP; and presented it as a generic set of ICT DPs.

This section proposes an SPECDP that is a specialized set of DPs. Specialized for building specific modules for ARS (or MRS); this SPECDP is applicable to a large variation of scopes, including very abstract fields, such as general robotic entities control or enterprises automation, as well as very concrete domains, in particular autonomous ARS or MRS.

An SPECDP is a solution to a ICS construct and maintenance problem in an ARS or MRS (or other context); each SPECDP describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that this solution can be reused.

SPECDPs are an attempt to describe successful specialized solutions to common ARS problems (such as communication vision control, communication management and other...).

The long term goal is to develop SPECDP handbooks for ARS Engineers (ARSE) (Adelson, B. & Soloway, E., 1985).

From one point of view, there is nothing new about the ARS SPECDP (or ACMDP in general) since by definition patterns capture experience. It has long been recognized that programmers do not think about programs in terms of programming language elements, but in higher-order abstractions (Soloway, E. & Ehrlich, K., 1984).

What is new is that ARSEs are working to systematically document abstractions other than algorithms and data structures. In general, most ARSEs working on patterns are not concentrating

on developing object oriented formalisms for expressing patterns for using them (Papurt, D., 1995), though a few of them are.

Instead, they are concentrating on documenting the SPECDP (or ACMDPs in general) that ARSE use. Relatively few ARSEs thoroughly understand and consistently apply SPECDPs (or ACMDPs in general) “like” DPs in their daily work (Linn, M. & Clancy, M., 1992).

In addition, advances on theoretical grounds for automated cognition show that this type of approach, managing complex systems in numerous small contexts, featuring well-structured, multiple levels of abstractions, is the most appropriate (Dessimoz, J.-D., 2000) (Dessimoz, J.-D., 2003).

The SPECDP proposes a set of DPs aiming to help ARS designers to build and forecast ARS problems, adjust the project management pathways and locate problem sources as well as to define their possible solutions.

Although we have many components and modules for designing and implementing most of the ARS software components (Gamma, E., Helm, R., Johnson, R. & Vlissides, J., 1995), until today we still do not have applicable SPECDPs in the areas of control and monitoring of complex ARS or an MRS, that is why we propose the ACMDP that can be added to the companies standard Design Pattern Catalog (DPC).

SPECDP (or ACMDP in general) is a recurring solution to a specialized ARS (or MRS) problem(s). This proposal is intended to be of interest for non-technical personal in charge of critical ARS systems, such as senior managers and auditors in the case of a company or the ARSE for the case of autonomous mobile robots (Trad, A. & Dessimoz, J.-D., 2004).

## 6. The Communication Design Patterns (COMMDP)

In general, Ethernet based standard « Transmission Control Protocol/Internet Protocol », which is called « TCP/IP », connects and uses a layered synchronous robust high speed communication protocol between any resources of the network that is connected to it.

In this section, we propose the « Personal Computer » (PC) based, «Communication Design Pattern » (COMMDP).

COMMDP is based on the TCP/IP protocol to support Remote communication, Control and Monitoring of «Various Delimiters»; such as ARSs and other automated production units.

This COMMDP by means of « TCP/IP » can optimally support, use and synchronize the plant production processes.

The COMMDP consists of modules that act as independent systems or components. The intention is to propose robust and low cost PC based

COMMDP solutions in order to effectively manage ARS, MRS or production units in a laboratory (or a plant); and to increase the productivity by linking various resources by means of TCP/IP and Ethernet (Trad, A., Gauthey, P-F., Loersch, M. & Dessimoz, J-D., 2004).

## 7. The Data Storage Design Pattern (DSDP)

This section presents a proposal of a DP “specialized” for modeling complex and voluminous « Multi-modal Object Data Sets » (MODS), focusing on the specific case of multimodal Audit, Control and Monitoring (ACM) information; with associated user-defined criterias (known also as primary features) and keys.

This DP is also used for: inter-tier data exchange, database manipulation and internal data presentation.

The system’s criterias understand the raw data, exceptions, application data, application scenarios, system statistics and logging.

The DSDP is a “high level” proposal of a specialized DP for modeling complex, voluminous and stream-oriented MODS; using XML technologies (Bourret R. 2001).

Essentially focusing on the specific case of multimodal application or ACM (XML based) data with associated user-defined criterias (known also as primary features) and keys; that the user can manipulate through a specialized Graphical User Interface (GUI) (Pardi, W. 1999).

The DSDP embeds the complex problem of global application data and the related behavior.

The « DSDP Model » is assessed using a large collection of entities of the captured design. For the ICS processing tasks, we look at predicting DSDP Quality Criterias (DQC).

This DSDP includes accessing fast-access multimedia database environments during the storage and modification operations, panic handling and control of business processes. The DSDP

Model is assessed using a large collection of annotated information of the ACM processes (Papurt, D. 1995).

For the annotation task, we look at predicting DQC on the manipulated specially formatted information (Fowler, M. & Rice, D., 2003).

These are some the basic requirements for the DSDP (Drake C. & Brown, k., 1995) (Trad, A., Gauthey, P.-F., Lüthi, C. & Dessimoz, J.-D., 2004).

## 8. The Vision Recognition Design Pattern (VRDP)

The VRDP proposes a construct for the support of ARS or MRS navigation, employing artificial

intelligence constructs (in this case it is the DMDP, that is presented in the next section).

The VRDP manages physical cameras and offers an abstraction layer to the captured image(s).

The VRDP is linked to the DMDP (two specialized DPs, which are part of the SPECDP) through the ACMDP; and that insures that the ARS (or the MRS) insure:

- ◆ Trajectory following.
- ◆ Collisions management.
- ◆ Object recognition.
- ◆ Obstacle avoidance in indoor environments.

Raw images of a specific size are processed one at a time by an image and position recognition algorithm (DMDP) (Trad, A. 1995).

The DMDP output signals control directly to the ARS’s motor control system.

## 9. The Decision Making Design Pattern (DMDP)

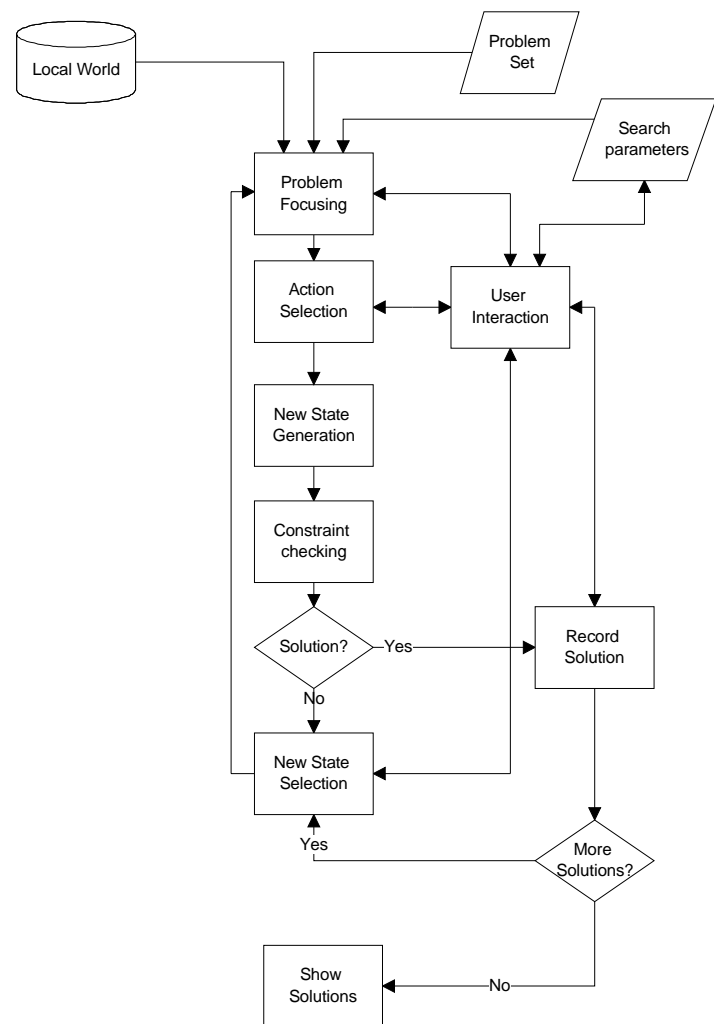


Fig. 2. Presents the DMDP flow diagram.

The main purpose of the DMDP (as shown in Fig. 2.) is to provide (i) a reasoning and the (ii) basis for a tool that supports the decision making process in various phases of ARS processing, auditing, monitoring and control.

The DMDP provides a quality support for dynamics of ARS control of activities (or tasks) that encompass project management planning issues (e.g. scheduling and rescheduling).

Complex relationships among execution activities (and tasks) or actions that can be undertaken in order to solve problems encountered before and during ARS processing, control, maintenance or auditing (which may themselves generate further problems).

DMDP exhibits an extensive dynamic nature; it can be tailored to be suitable in various problem domains, i.e. any real world environment.

This includes finding the best solution for the current ARS problem(s).

However the user can (a) tackle a single or multiple problems, (b) which could be dealt with by DMDP, (c) solutions are suggested by DMDP (d) which in turn are selected or rejected by the user.

All encountered problems and a set of their corresponding solutions are stored and categorised within the “ARS task description database” known also as the “ARSDB”.

However the DMDP should find the best solution for any detected problem, hence DMDP searches for all possible solutions using the “Beam Search Algorithm”.

It is very often the case that a selected solution can solve a range of more common types of problems, but its selection could be based on experienced ARSE and therefore includes some of the “tricks” excerpted from previous experiences.

Once the solution search procedure (RSRPROC) is initiated, the DMDP system should be capable in a short period of time to provide a whole set of solutions (if any?).

The ARS (or ARSE if the system is offline) will have the possibility of choosing one of these solutions and implementing it (which might include rescheduling the ARS execution scenario, control, maintenance, planning tasks if necessary) (Trad, A. 1995).

## 10. The Human Robot Interaction Design Pattern (HRIDP)

The most desired properties of the ARS or the MRS system are as follows:

- ◆ The ARS or the MRS should adapt to numerous variations in its environment and yet reach a pre-assigned goal.

- ◆ The ARS or the MRS must be capable of low-level reactive behavior as well as higher level cognitive performance (re. cognitics)
- ◆ The ARS or the MRS should be “user-friendly”, and accept very fast and easily new instructions.
- ◆ The user can discover the ARS or the MRS capabilities through a Graphical User Interface (GUI). This implies that it has a “user-friendly” graphical user interface as shown in “Fig. 3”.
- ◆ The ARS or the MRS can detect its failures (by interacting with the DMDP and VRDP), possibly continue with a lower-level of expertise (with the help of an external monitoring tool such as FES, as show in “Fig. 5.”), and inform the ARSE for assistance.

A lot of information can also be acquired visually. The ARS or the MRS VRDP manage vision through embedded cameras. In the market there is a wide range of cameras that can be used; currently there are two major groups of interest for us: 1) USB and 2) HTTP/TCP driven cameras.

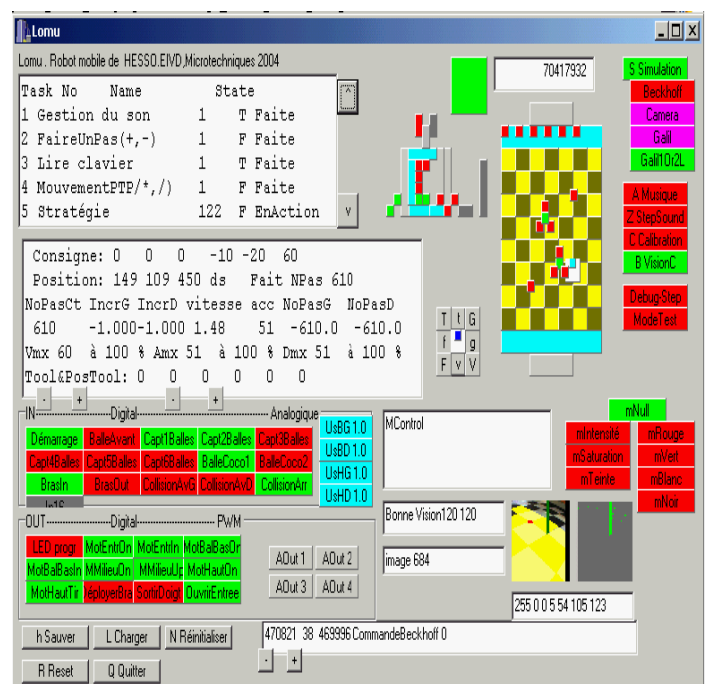


Fig. 3: Human Robot Interaction (HRI) is made easier with graphic, real-time e display and simulation properties, mouse-based and control-key functions; multiple real-world oriented debugging aids, as well as high abstraction, hyper parallel command language.

## 11. The FES Design Pattern (FESDP)

Once the ARS or the MRS is aware of the presence of another object in its environment, the monitoring module is responsible for keeping up with the object and all that is known about it. This

includes monitoring physical characteristics as well as information inferred from the object. Localization and tracking algorithms allow the ARS or the MRS to keep up with the object's place in space.

“Watching” the object includes tracking visual characteristics of objects. The monitoring module is responsible for scheduling and monitoring the data from the tracking. The monitoring module also monitors object communication and interprets input from the object to determine if there is important information being conveyed (Trad, A. & Kalpic, D., 2004).

### 11.1 What are factors?

This section presents the FESDP (and other ACMDP patterns) direct and indirect *factors* that directly influence the success of ARS auditing, development, monitoring, control and reengineering.

The FESDP (and other ACMDP patterns) has a pool of *factors* that play a role of major phenomena and determine the ARS (or MRS) quality status and characteristics.

A *factor* identifies the type of problem, which in turn results in corresponding action(s).

The FESDP (and other ACMDP patterns) factors are independent of all the other ACMDPs patterns or views.

### 11.2 What are views?

GLOBAL VIEW	
The TEAM VIEW	(TEAMV)
The TECHNICAL VIEW	(TECHV)
The DOMAIN VIEW	(DOMNV)
The SECURITY VIEW	(SECUV)
The SYSTEM DESIGN VIEW	(SYSDV)
The ORGANISATIONAL VIEW	(ORGAV)
The FINANCIAL VIEW	(FINAV)
The LEGAL VIEW	(LEGAV)
The POLITICAL VIEW	(POLIV)
The AUDIT VIEW	(AUDV)

Fig.4. The system views.

A view is defined as a set of one or more factors. Hence, if  $f_{ij}$  is a factor and  $V_j$  is a view which selects factor  $f_{ij}$  we could define:

- ◆  $f_{11}, \dots, f_{1r}$  determines View  $V_1$
- ◆  $f_{21}, \dots, f_{2s}$  determines View  $V_2$
- ◆ .....
- ◆  $f_{m1}, \dots, f_{mt}$  determines View  $V_m$

Sets  $F_i$  and  $F_j$  of factors  $f_{i1}, \dots, f_{in}$  and factors  $f_{j1}, \dots, f_{jm}$  are not necessarily disjoint sets.

The creation of different views (as shown in figure 4), i.e. group factors, results in different categories of factors.

### 11.3 Proactive monitoring

Proactive monitoring using the FES is essential because every ARS or the MRS, with very few exceptions, is far too complicated to be managed without some kind of automated assistance. Of course the main requirement is autonomy! In fact, automation is implied in the term proactive monitoring (Trad, A., Kalpic, D. & Fertalj, K. 2002).

Monitoring an ARS or the MRS manually or with ad hoc modules, without the help of FESDP or other software tools, is reactive rather than proactive.

FESDP can monitor multiple ARS or the MRS. The system was built using the GIDP MVC pattern (MVCDP) (Von Zimmermann, P, 1992).

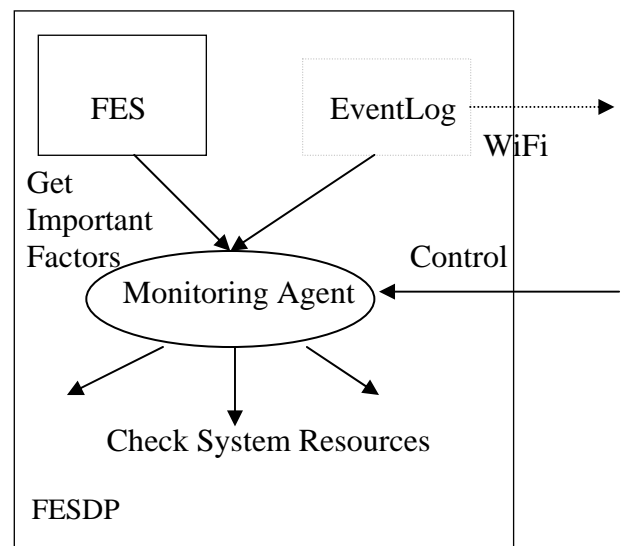


Fig.5. The FESDP.

Monitoring can take the form of simple scripts or system programs that are invoked through system schedulers, or it can involve complex packages like the FESDP as shown in “Fig. 5”.

No matter how much computer power or software the support team throws at the system monitoring, some kind of manual intervention will be always required, with only few exceptions such as auto restart and auto recovery routines.

The analogy to a tree in the forest that falls without anyone hearing is a good one. If no one reads the script-generated reports, looks at the monitoring station's graphical display, or listens to the incoming alarms, is the system really being monitored at all? Of course, not!

One can say that such a system is not even maintained. For the FESDP, therefore, the term proactive monitoring implies several *factors*: automation, detection of problems before they become critical, and reception of reports and alarms by someone who understands their significance (Trad A. & Kalpic, D., 1999).

Proactive monitoring is not something that should be cavalierly put aside until there is nothing else on the development team's schedule to be done (Trad, A., & Kalpic, D., 1998).

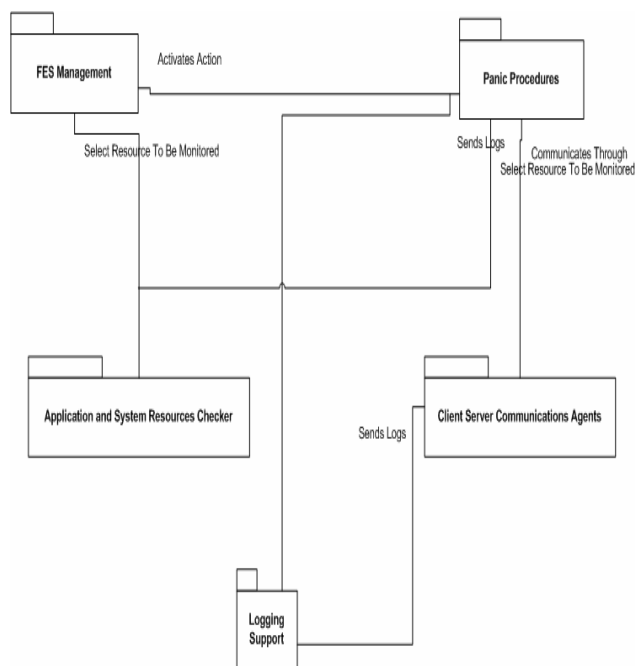


Fig. 6. The FESDP's main packages.

As soon as the ARS or an MRS is put into production, it is time to start to maintain and protect it against all possible types of problems.

The following questions must be asked when integrating the FESDP (as shown in "Fig. 6.") for an ARS or an MRS:

- ◆ What kind of problems thus *factors* are possible?
- ◆ Which of these problems, hence factors, can cause serious disruption of the ARS or an MRS?

- ◆ What information needs to be known about the status of the ARS or an MRS to keep it up and running?
- ◆ When do the support personnel (auditors) need to know about the problems?
- ◆ How is the support team notified?
- ◆ What are the panic procedures (Drake C. & Brown, k., 1995)?

The previous issues should be fully managed by the FESDP.

This section suggests the application of FESDP that can provide useful mechanisms to audit control and monitor and notify the eminence of problem occurrence (Jones, 1996).

Notice that the FESDP can be built on a 3<sup>rd</sup> party basic monitoring infrastructure, for example IPmonitor (IPmonitor, 2004) or Tivoli (Tivoli, 2004).

With respect to the activities and *factors*, the following requirements, typical for mobile robots, can be stated for the delivery of the FESDP:

- ◆ **Collisions:** The FESDP must be capable of foreseeing collisions, and of repositioning the ARS or an MRS when that happens.
- ◆ **Performance:** The focus is on the "Operational Core" specification with respect to the performance of basic tasks. Each basic task must be performed within a precise time sequence that depends on the type of the task. For instance, providing raw materials is a task that must be performed before the production process begins.
- ◆ **Integrity checking:** Checking the plans validation and resilience.
- ◆ **Filtering:** Reducing the amount and complexity of data that one needs to review. Examples include log files, error messages, quota reports, and the output from scheduled jobs.
- ◆ **Health checking:** Attaching probes to essential services so that one is notified immediately when they become unavailable. Examples of services one needs to monitor in this way include web servers and email servers (Krishnamurthy, B. & Rexford, J., 2001). These checks can be also applied to ARS or an MRS ethernet cameras that have embedded web servers.
- ◆ **Autonomy:** Insuring the system's autonomy.
- ◆ **Connection stickiness:** Maintaining connectivity and performance statistics, through the use of graphical displays of the network being monitored. No service is useful if the system on which it is running cannot be accessed.
- ◆ **Panic procedures:** Assisting in the detection of faults, using emergency procedures (Drake C. & Brown, k., 1995).



requirements; these all are examples of problem detections before they disrupt the workflow of an ARS or an MRS. This set of defined factors is known as the ARS or an MRS (failure) Factors Estimation Set (FES, please do refer to section -6- for more information on FES and FESDP).

*The ACM DP*, can be adapted to any monitoring system. A good place to begin is to identify the possible sources of problems (known as factors), which in the ARS or an MRS monitoring system are known as factors (Trad, A., Kalpic, D. & Fertalj, K. 2002).

### 13. The Underlying Infrastructural Architecture

Client/Server (CS) computing has created a deep paradigmatic shift in the ICS industry; It's replacing colossal monolithic mainframe systems with applications split across different computation units; in a client and server environment also known as an n-tier system (Trad, A., 2001).

Stateless business objects in the form of XML strings are a paradigm shift within a paradigm shift; this is a new client server revolution within the client/server revolution. Stateless business objects in XML format break-up the client and server sides of an application into independent components that can interact together and roam across networks, with a unique and flexible interface definition (Trad, A. & Kalpic, D., 2001).

#### 13.1 XML Based Systems

XML is an open, text-based mark-up language that provides structural and semantic information to data. XML, a subset of the popular Standard Generalised Mark-up Language (SGML), has been optimised for the Web. This makes XML a powerful, standards-based complement to HTML that could be as important to the future of information delivery on the Web as HTML was to its beginning.

XML is intended to be used by content creators as well as by programmers. Since XML is text-based, it can be read and worked with easily in relatively non-technical situations, but its ability to organise, describe, and structure data also makes it ideal for use in highly technical applications. XML thus provides common ground for creating structured data and making it available for manipulation and exchange between the different tiers (Pardi, W., 1999).

Today the client/server computing systems are replacing the colossal monolithic mainframe systems with components installed on different computation units (clients and servers); due to that the components interface definitions and

dependencies became a major problem (Orfali, R., Harkey, D. & Edwards, J., 1996).

That's why the stateless business objects formatted in XML strings, brought a solution to this serious problem and made an enhancement to the previous client/server technology standards. Stateless business object in XML format, break-up the client and server sides of an application into independent components that can interact together and roam across networks. This interaction is done through one single interface call, thus the frequent data model changes do not imply component interfaces modifications (Albrecht, K., 2000).

#### 13.2 System's Modeling

One of the main aims of building such a system is "to design it once and to use the resultant design on all three tiers" (Innovator, 2000).

To achieve that goal it is very important to respect the "1:1" rule through all three tiers; this rule notes that UML the design (Burkhardt, R., 1997) documents must be identical and stay unchanged between the different tiers.

This was fully achieved for the data modelling part, whereas for the behaviour modelling (Eriksson, H E. & Magnus P., 1998) it was not possible to achieve that goal.

Because two different development environments were used on the 1<sup>st</sup> tier's client and the 2<sup>nd</sup> tier's server.

XML is rapidly establishing itself as the meta-grammar for inter-organisational communication around the Internet. It is becoming increasingly urgent that business analysts, systems analysts, and software developers are able to:

- ◆ Model the information to be represented in XML (data modelling)
- ◆ Describe the relationships between the XML and the systems to process it (behavior modeling)

Having done so, they must also be able to rapidly generate the boilerplate code associated with the implementing these processes. At present there is no tool capable of doing this (Booch, G., Christerson, M., Fuchs, M. & Koistinen, J., 1999).

#### 13.3 The Object Oriented to Relational Database mapping issues

The object paradigm is based on software engineering principles such as coupling, cohesion, and encapsulation, whereas the relational paradigm (El Masri & R., Navathe, S., 1994) is based on mathematical principles, particularly those of set theory.

The two different theoretical foundations lead to different strengths and weaknesses. Furthermore, the object paradigm is focused on building applications out of objects that have both data and behavior, whereas the relational paradigm is focused on storing data.

The "impedance mismatch" comes into play when you look at the preferred approach to access: with the object paradigm, you traverse objects via their relationships, whereas with the relational paradigm you duplicate data to join the rows in tables. This fundamental difference results in a less-than-ideal combination of the two paradigms, but then, a few hitches are to be expected. One of the secrets of success for mapping objects to relational databases is to understand both paradigms and their differences, and then make intelligent trade-offs based on that knowledge (Ambler, S., 2000).

The Object Identifiers (OIDs) and the Unique Object Identifiers (UOID) are used to identify objects and in relational databases they are used as primary keys (Ambler, S., 2000).

The Object Oriented (OO) to Relational DataBase Management System (RDBMS) system mapping layer was built to tackle the following issues:

- ◆ Name mapping filter, for various reasons such as naming conventions, the 2<sup>nd</sup> and 3<sup>rd</sup> tier's data source systems can have different names to identify the same attributes.
- ◆ Performance, as the system uses the dynamic SQL motor to access relational databases that can cause major performance degradations. Most relational database systems have configuration parameters that set-up the 'cache' in order to enhance this type of performance issue. For example the IBM DB2 has the 'Paragraph' parameter for that issue.
- ◆ Referential integrity and OO integrity, which integrity should be taken as base? (LPC Consulting, 1998)

Database editors (like Oracle) propose extenders for the OO to relational map (Chang, B., Scardina, M., Karun, K., Kiritzov, S., Macky, I., Novoselsky, A. & Ramakrishnan, N., 2001).

## 14. ACMDPs System's Architecture

Our ARS or an MRS is a distributed system built with real-time agents, and is designed to run under MS Windows operating systems; whereas it can be easily ported to Unix based operating systems (Stapleton, J., 1997).

As it is written mainly in Borland C++ (Kolachina, S., 2000) (Hollingworth, J., Gustavson, P., Swart, B. & Cashman, M., 2002) and C#.NET was used for the ACMDP interface (Telles, M., 2002) (Hoque, R., 2000) (Templeman, J. & Olsen, A.,

2002), all the components communicate using the TCP/IP supported network.

ARS or an MRS architecture, also known as Multi-Agent System, specify how to accomplish and integrate planning, monitoring, and control into a system constrained by: (1) sensors and actuators, which cause failures and misperceptions; (2) restricted computational resources, which limit the amount of planning and data processing; and asynchronous events (Trad, A., Kalpic, D. & Fertalj, K. 2002), such as failures or errors, which call for immediate attention; and (3) probably the most important hyper-fast processing (In average, it takes about 100 nanoseconds for an agent to take control and perform an elementary operation) (Dessimoz, J.-D., Gauthey, P.-F. & Roulin, P., 2003).

System architectures describe the software components at a macro level in terms of a manageable number of packages inter-related through data and control dependencies (Land, R. & Crnkovic, I., 1999).

The design of such software architectures has been the focus of considerable research for the past decade, which has resulted in a collection of well understood architectural styles and a methodology for evaluating their effectiveness with respect to particular software requirements (Stapleton, J., 1997).

Examples of software qualities include maintainability, modifiability, portability, etc. ARS or MRS architectures can be considered as libraries of components composed of autonomous and proactive agents that interact and cooperate with each other in order to achieve common or private goals.

In this paper we propose a very basic architecture to design the architecture (Zimmermann, P, 1992) of ARS or MRS components and we do not compare it to conventional architectural solutions. The problem focuses on embedded real-time or hyper fast systems.

The ACMDP must deal with external sensors and actuators and must respond in time constants commensurate with the activities of the system in its environment. In our case it is being developed in tight synchronisation with a new, proprietary programming language, which offers unmatched parallelism options.

An autonomous system typically has to accomplish the following operations: acquiring the input provided by sensors, controlling the motion of wheels and other moving parts, and planning of future trajectories.

In addition, a number of factors complicate the tasks: obstacles may block the ARS or an MRS's path, sensor inputs may be imperfect, the ARS or an MRS may run out of power, mechanical limitations may restrict the accuracy with which the ARS or an MRS moves itself, the ARS or an MRS

may manipulate hazardous materials, unpredictable events may leave little time for responding (Hasemann, J-M., 1996).

In the ARS or an MRS's structure, checks and control mechanisms can be integrated at different abstraction levels ensuring redundancy from different perspectives, such as availability, mirroring or even load-balancing.

Contrary to the conventional architecture, checks and controls are not restricted to adjacent layers. Besides, since the structure must permit the separation of the data and control hierarchies, integrity of these two hierarchies can also be verified independently. The joint venture, through its joint manager, proposes a central message server.

The exception and error-management mechanisms, such as wiretapping or supervising can be supported by the joint manager. This manager guarantees non-fallibility, reliability and completeness by the use of robust panic procedures. The most important (key) architectural requirements in engineering the ACMDP packages are:

- ◆ **Predictability:** Agents can have a high degree of autonomy in the way they undertake action and communication in their domains. Then it can become difficult to predict individual characteristics as part of determining the behaviour of the system as a whole. For an ARS or an MRS, all the circumstances of the operations will never be fully predictable. The architecture must provide the framework in which an ARS or an MRS can act even when faced with incomplete or unreliable information, like for example contradictory sensor readings.
- ◆ **Fallibility-Tolerance:** The failure of a single agent does not necessarily imply a failure of the whole system. Then the system needs to check the completeness and the accuracy of data, information and transactions. To prevent from system failure, different agents can, for instance, implement replicated capabilities. The architecture must prevent the failure of the ARS or an MRS's operation and the environment. Local problems, like reduced power supply, dangerous vapours, or unexpectedly opened doors should not necessarily imply the failure of a mission.

In this paper we propose an ACMDP and the possible types of architectures are (Eriksson, H E. & Magnus P., 1998):

- ◆ **Conventional Architectures** serve for simple classical solutions. Due to lack of space, we only examine the four major conventional architectures - the layered architecture, the

distributed, control loops and task trees that can be implemented in an ARS or an MRS.

- ◆ **Layered Architecture** is a seven level architecture, where in its lowest level (level 1) reside the ARS or an MRS control routines (motors, joints ...). Levels 2 and 3 deal with the input from the real world. They perform sensor interpretation (the analysis of the data from a single sensor) and sensor integration (the combined analysis of different sensor inputs). Level 4 is concerned with maintaining the ARS or an MRS's model of the world. Level 5 manages the navigation of the ARS or an MRS. The next two levels, 6 and 7, schedule and plan the ARS or an MRS's actions. Dealing with problems and re-planning is also part of level 7 responsibilities.
- ◆ **Distributed architecture** based on computing systems with components installed on different computation units (clients and servers) is replacing the once prevailing colossal monolithic mainframe systems. The component interface definitions and dependencies are becoming a major problem. A stateless business object in XML format breaks-up the client and server sides of an application into independent components that can interact and roam across networks. This interaction proceeds through one single interface call, thus the frequent data model changes do not imply component interface modifications (Hoque, R., 2000).
- ◆ **Control loop** is a controller component to initiate the ARS or an MRS actions. Since an ARS or an MRS has responsibilities with respect to its operational environment, the controller also monitors the consequences of the ARS or an MRS actions adjusting the future plans based on the feed-back information.
- ◆ **Task Trees architecture** is based on hierarchies of tasks. Where parent tasks initiate child tasks.

In addition, R. Brooks's subsumption architecture is also a good answer for some niche, namely the reactive behaviour of animates in situation.

## 15. ACMP Maintenance

Concerning maintenance, this section proposes the possible application of a monitoring component, where control and monitoring operations may (and frequently do) result in a reengineering process. Once the ARS or an MRS is put into production, the maintenance and reengineering considerations start, and the monitoring component should be periodically used to control it.

That is an important factor in the ARS or an MRS lifecycle; it may result in a reengineering process that in most cases is due to serious problems and



This paper's proposal is made in the form of an ACM DP requirement description that can also be applied to other autonomous and complex systems. This paper also includes a set of component requirements that are relevant to this ACM DP.

Future research directions include formalizing precisely the interactive and proactive structures that have been identified, as well as the sense in which a particular model is treated as an instance of such a style and pattern.

We also propose to relate them to existing patterns and lower-level architectural components involving software components, ports, connectors, interfaces, libraries and configurations.

We are still working on contrasting our structures to conventional styles and patterns proposed in the software engineering literature.

## 18. References

- Adelson, B. & Soloway, E. (1985). The Role of Domain Experience in Software Design. *IEEE Trans. on Software Engineering*, V SE-11, N 11, pp. 1351-1360.
- Albrecht, K. (2000). Submission and award, for Object Management Group Applications Awards 2000, Winterthur e-Business department, Winterthur, Switzerland.
- Ambler, S. (2000). Mapping objects to relational databases, Ronin International-IBM developerWorks, USA.
- Booch, G., Christerson, M., Fuchs, M. & Koistinen, J. (1999). UML for XML Schema Mapping Specification, Rational Rose, USA.
- Bourret R. (2001). Mapping DTDs to Databases.
- Brown L. (1997). The New Shorter Oxford – English Dictionary – Volume I, Clarendon Press, Oxford, Oxford University Press, UK.
- Burkhardt, R. (1997). UML Unified Modelling Language. USA: Addison Wesley.
- Chang, B., Scardina, M., Karun, K., Kiritzov, S., Macky, I., Novoselsky, A. & Ramakrishnan, N., (2001). Generating XSQL Server Pages, Oracle Magazine-Oracle Press, USA.
- Derrien, Y. (1992). Les techniques de l'audit informatique. France: DUNOD.
- Dessimoz, J.-D. (2000). Beyond information era : cognition and cognitics for managing complexity; the case of enterprise, from a holistic perspective", Proc. ICIMS-NOE (Intelligent Control and Integrated Manufacturing Systems- Network of Excellence), ASI - 2000 (Advanced Summer Institute) and Annual Conference, CNRS-ENSERB-Université de Bordeaux, Bordeaux, France , 18-20 Sept. 2000, pp.164-170
- Dessimoz, J.-D. (2003). Knowledge Management and Cognitics; Some Fundamental Aspects" Proceedings « Workshop IPLnet 2002 ; From Research to Application », IPLnet Swiss National Network of Excellence in Intelligent Manufacturing and Logistics, p. a. École d'Ingénieurs du canton de Vaud , HES-SO, Yverdon-les-Bains, Febr. 2003.
- Dessimoz, J.-D., Gauthey, P.-F. & Roulin, P. (2003). Is a Robot that can autonomously play soccer intelligent?; Hesso/Eivd - West Switzerland University of Applied Sciences, 1400 Yverdon-les-Bains, Switzerland.
- Dessimoz, J.-D. (2004). Project Recommendation – MRS Swiss Robotics Cup; Hesso/Eivd - West Switzerland University of Applied Sciences, 1400 Yverdon-les-Bains, Switzerland.
- Dayton, D. (1999). Information Technology Audit Handbook. USA: Daytonassociates.
- Drake C. & Brown, k. (1995). PANIC! UNIX System Crash Dump Analysis Handbook. NJ USA: Prentice Hall Inc..
- El Masri & R., Navathe, S. (1994). Fundamentals of Database Systems, World Student Series, CA, USA.
- Eriksson, H E. & Magnus P. (1998). UML Toolkit. USA: John Wiley & Sons, Inc.
- Fowler, M. & Rice, D. (2003). Patterns of Enterprise Application Architecture, Addison-Wesley, USA.
- Gamma, E., Helm, R., Johnson, R. & Vlissides, J. (1995). Design Patterns. 395 pages, Addison-Wesley Professional; 1st edition; ISBN: 0201633612.
- Hasemann, J-M. (1996). A control architecture for intelligent robots based on graph manipulation for planning, monitoring, execution, intention switching, and fault recovery; Technical Research Center of Finland, 1996, VTT Publications.
- Hollingworth, J., Gustavson, P., Swart, B. & Cashman, M. (2002). Borland C++ Builder 6 Developer's Guide, SAMS, USA.
- Hoque, R. (2000). XML for Real Programmers, Academic Press, San Diego, CA, USA.
- Innovator, (2000). INNOVATOR CASE - strategic software development tool, MID Corporation, Germany.
- IPmonitor (2004). The IPmonitor Web Monitor website, <http://www.ipmonitor.com/>
- ISACA-S (1998). Standards For Information Systems Auditing, USA: ISACA.
- Lewis, R. (2004). Modeling control systems using IEC 61499, The Institution IEEE International CONFERENCE of Electrical Engineers ON MECHATRONICS, ISTANBUL TURKEY, 2004.
- Linn, M. & Clancy, M. (1992). The Case for Case Studies of Programming Problems. Communications of the ACM V 35 N 3, March 1992, pp. 121-132.
- Magnus H.E. & Penker M. (1998). UML Toolkit, Wiley computer publishing, USA.

- Meier, R. (1997). Team Power, Die Deutsche Bibliothek - CIP Einheitsaufnahme, Germany.
- Mobile Robot LOMU (2004). [www.robot-ch.org](http://www.robot-ch.org)
- Nance R. & Arthur J. (1988). The methodology roles in the realization of a model development environment, Proceedings of the 20th conference on Winter simulation, San Diego, California, United States, Pages: 220 - 225, ISBN:0-911801-42-1.
- Jones (1996). Applied Software Measurement: Assuring Productivity and Quality, USA: McGraw-Hill.
- Kolachina, S. (2000). C++ Builder 6 Developer's Guide, Wordware Publishing, Inc, USA.
- Krishnamurthy, B. & Rexford, J. (2001). Web Protocols and Practice: HTTP/1.1, Networking Protocols, Caching, and Traffic Measurement. USA: Addison-Wesley Pub Co.
- Land, R. & Crnkovic, I. (1999). Software Systems Integration and Architectural Analysis – A Case Study , Proceedings of International Conference on Software Maintenance (ICSM), Amsterdam, Netherlands, 2003.
- LPC Consulting (1998). OO to RDBMS Mapping Framework, LPC Consulting Services Inc. USA.
- Orfali, R., Harkey, D. & Edwards, J. (1996). The Essential Distributed Objects-Survival Guide, John Wiley & Sons Inc., Canada.
- Pardi, W. (1999). XML in Action, Microsoft Press, Redmond, Washington, USA.
- Papurt, D. (1995). Inside the Object Model - The Sensible Use of C++. USA: Sigs Books.
- Robotics Cup (2004) <http://robotik.com/cup/cup.html>
- Selic, B. & Rumbaugh, J. (1999). "Using UML for Modeling Complex realtions of the ACM", October 1999, Vol. 42, No.10. Time Systems", Rational Software, 1998.
- Simoneau, P. (1999). SNMP Network Management. USA: McGraw-Hill Computer Communications Series.
- Soloway, E. & Ehrlich, K. (1984). Empirical Studies of Programming Knowledge, IEEE Transactions on Software Engineering V SE-10, N 5.
- Stapleton, J. (1997). Dynamic Systems Development Method. The method in practice (DSDM), DSDM Consortium 1997, Addison Wesley Longman Limited, UK..
- SwissCup (2004). Swiss Cup, Yverdon-les-Bains, Switzerland.
- Telles, M. (2002). C# Black Book, The Coriolos Group, LLC., USA.
- Templeman, J. & Olsen, A. (2002). Microsoft Visual C++ .NET, Microsoft Corporation, USA.
- Thramboulidis, K. (2004). Using UML in Control and Automation: A Model Driven Approach", Paper, INDIN 2004, Berlin, Germany.
- Tivoli (2004). The TIVOLI monitoring system: <http://www-306.ibm.com/software/tivoli/>
- Trad, A. (1995). Intelligent Rescheduling System (IRS) For Operations Control, SairGroup, CH.
- Trad, A. (2001). An Optimal Client/Server Architecture Using Java, C++ and XML. TradSoft. Java Days 2002; Zurich, Switzerland.
- Trad, A. (2004). Workshop Presentation "Swiss Robotics Cup"; Hesso/Eivd - West Switzerland University of Applied Sciences, 1400 Yverdon-les-Bains, Switzerland.
- Trad, A. & Swissair (1997). The Absolute Monitoring Method – AMM, Kloten, Switzerland: Swissair, SairGroup.
- Trad, A. & Dessimoz, J.-D. (2004). An Architectural Design Pattern Proposal for Mobile Robot Systems; INDIN 2004, Berlin, Germany.
- Trad, A. & Dessimoz, J.-D. (2004). Proactive Monitoring and Maintenance of Intelligent Control Systems - Application for Mobile Robot Systems and Collaborative Project Management Environment, IMS-NoE International Conference. Cernobbio, Italy.
- Trad, A., Gauthey, P.-F., Loersch, M. & Dessimoz, J.-D. (2004). PC based Distributed Systems Control for Production, Automation - Ethernet and TCP/IP based solutions, IPLnet 2004, Morat, Switzerland.
- Trad, A., Gauthey, P.-F., Lüthi, C. & Dessimoz, J.-D. (2004). Cognitive Processing of Multimodal Images in Large Databases Environments with Fast Access A Multi-modal Object Management Design Patten (MOMDP) PART I - "The Overview", IPLnet 2004, Morat, Switzerland.
- Trad, A., Gauthey, P.-F., Lüthi, C. & Dessimoz, J.-D. (2004). Cognitive Processing of Multimodal Images in Large Databases Environments with Fast Access; A Multi-modal Object Management Design Patten (MOMDP) - PART II - "The XML Based Framework", IPLnet 2004, Morat, Switzerland.
- Trad, A., & Kalpic, D. (1998). The Team Design Quality Check Coefficient (TDQCC). Proceedings of the International Conference on Information Technology Interfaces; 1998 Jun 1998; Pula, Croatia. Zagreb: SRCE University Computing Center, University of Zagreb; 1998. p. 359-364.
- Trad A. & Kalpic, D. (1999). Reengineering Quality and Risk Check (RQRC) - A Practical Implementation. Proceedings of the 24<sup>th</sup> International Conference on Information Technology Interfaces; 1999 Jun 15-18; Pula, Croatia. Zagreb: SRCE University Computing Center, University of Zagreb; 1999. p. 491-496.

- Trad A. & Kalpic, D. (1999). Reengineering Quality and Risk Check (RQRC) – Theoretical Basis. Proceedings of the 24<sup>th</sup> International Conference on Information Technology Interfaces; 1999 Jun 15-18; Pula, Croatia. Zagreb: SRCE University Computing Center, University of Zagreb; 1999. p. 497-502.
- Trad, A. & Kalpic, D. (2001). BUILDING A XML BASED OBJECT MAPPING SYSTEM (OMS) – Proceedings of the International Conference on Information Technology Interfaces; 1998 Jun 2001; Pula, Croatia. Zagreb: SRCE University Computing Center, University of Zagreb; 2001.
- Trad, A., Kalpic, D. & Trad, C. (2002). Applying the ISRQCC Method in a Web Reengineering Process - The SwissInsurances (SWI) Web Engineering Audit, Proceedings of the 26<sup>th</sup> International Conference on Information Technology Interfaces; 2002 Jun 24-27; Dubrovnik, Croatia. Zagreb: SRCE University Computing Center, University of Zagreb; 2002.
- Trad, A., Kalpic, D. & Fertalj, K. (2002). Proactive monitoring of the information system risk and quality. Proceedings of the 26<sup>th</sup> International Conference on Information Technology Interfaces; 2002 Jun 24-27; Dubrovnik, Croatia. Zagreb: SRCE University Computing Center, University of Zagreb; 2002. p. 497-502.
- Trad, A. & Kalpic, D. (2004). Proactive Monitoring and Maintenance of Intelligent Control Systems: A Design Pattern for Autonomous Systems. ITI 2004, Dubrovnik, Croatia.
- Trad, A., Perrenoud, A., Gauthey, P-F. & Dessimoz, J.-D., Marcuard, J.-D. & Riedi, M. (2004). Improving Automated Manufacturing using Robots, AMEM - A Proposed Methodology. IPLnet 2004, Morat, Switzerland.
- Trad, A. & Swissair (1996). The Message Management System – MMS. Kloten, Switzerland: Swissair, SairGroup.
- Trad, A. & Swissair (1997). Monitoring Method – AMM, Kloten, Switzerland: Swissair, SairGroup.
- Von Zimmermann, P (1992). Das MVC-Modell, Germany: SAP AG.
- Vogel, A., Vasudevan, B., Benjamin, M. & Villalba, T., (1999). C++ Programming with CORBA. Canada: John Wiley & Sons Inc..
- Watters, P. & Veeraghawa (2000). Solaris–The Complete Reference. USA: McGraw-Hill.
- [www.rational.com](http://www.rational.com) (2004).