Limits to Ground Control in Autonomous Spacecraft

Alfred D. M. Wan, Peter J. Braspenning and Gerard A. W. Vreeswijk

University of Maastricht, Department of Computer Science
P.O. Box 616
6200 MD Maastricht, The Netherlands

Tel. +31 43 88[2023], [3491], [2021].
e-mail: [wan], [braspenning], [vreeswyk]@cs.rulimburg.nl

Abstract

In this paper the autonomy concept used by ESA and NASA is critically evaluated. Moreover, a more proper ground-control/spacecraft organizational structure is proposed on the basis of a new, more elaborated concept of autonomy. In an extended theoretical discussion its *definitional* properties and functionalities are established. The rather basic property of adaptivity leads to the categorization of behaviour into the modes of *satisfaction* and *avoidance* behaviour. However, the autonomy property with the most profound consequences is *goal-robustness*. The mechanism that implements goal-robustness tests newly generated goals and externally received goals on consistency with high-level goals. If goals appear not to be good instantiations or more acceptable replacements of existing goals, they are rejected. This means that ground-control has to *cooperate* with the spacecraft instead of (intermittently) commanding it.

1 Introduction

In current spacecraft control engineering two theoretical approaches can be distinguished, viz. [1] equipping spacecraft with autonomy, making them less dependent on ground control and [2] distributing intelligent functions to optimize performance on pre-defined system requirements. Spacecraft autonomy is viewed as a major design goal by leading institutions such as NASA and the European Space Agency (ESA). The main reason for making spacecraft less dependent on ground control, cf. [1], is that total control of the spacecraft is practically unfeasible due to e.g. limited visibility of on-board events. The reason for distributing intelligent functions, cf. [2], is optimization from the point of view of the complete ground-control/spacecraft organization e.g. reduction of operation costs and localization of computational resources, cf. (Grant, 1994; Aarup et al., 1994).

Usually autonomy is defined loosely, which inevitably leads to problems when it is attempted to be used as a design specification cf. (Easter & Staehle, 1984). We critically evaluate the autonomy concept developed by ESA in section 2. From this it will be clear that, before trying to use it, the concept of autonomy needs to be defined first, which is the aim of this paper. The concept of autonomy is developed from contrasting two possible organizational design stances as known in Distributed Artificial Intelligence (DAI), viz. Multi-Agent Systems (MAS) and Distributed Problem Solving (DPS) in section 3, that correspond to the design stances [1] and [2]. In MAS and [1] the emphasis is on autonomy, while in DPS it is on dividing and localizing the functionality of the whole system. It will be pointed out that the functionality of autonomy and the property of independence[1] belong to MAS. Although DPS and MAS may be seen as poles of a continuum, the predominant pole determines both the agent architecture and the organizational possibilities.

In section 4 and 5 we will engage in a full discussion of the origin of autonomy and its functionality. The argument runs as follows: agents that are exposed to uncertain circumstances in which they want to persist have to be adaptive. Systems theory provides an elementary architecture that is maximally adaptive (a feedback system) but has one fundamental inability: it can't change its own goals. Yet, an agent that is based on a feedback architecture can generate or receive new goals, but they have to be instantiations of the unchangeable, high-level goals. In this respect changes to, or generation of goals is restricted, which provides a heuristic warranty to goal approach; this is called *goal robustness.* Goal robustness provides independence from other agents, it will only commit itself to goals that conform with its high-level goals. Independence is thus specified and is a major characteristic of autonomous agents. Finally, we will evaluate what the application in spacecraft architecture of this newly developed concept of autonomy would mean.

2 Spacecraft Autonomy and Automation: a Critical Evaluation

The autonomy concept as developed by ESA (the Standard Generic Approach to Spacecraft Autonomy and Automation; SGASAA, cf. (Pidgeon et al., 1992) was primarily intended to enable spacecraft to continue with their mission, in case of *temporary* loss of contact with ground control. Any spacecraft that can't be controlled from the ground station and has no means of controlling itself soon perishes. Additional motivations for making spacecraft more independent from ground control, are that due to small communication bandwidths of deep space missions there is little visibility of on-board events and, additionally, long transmission time weaken promptness of the spacecrafts reaction. Also, operation costs would be significantly reduced because there is no need for continuous presence of "marching armies" of ground controllers[2].

[1] Independence is often equated with autonomy, as in (Easter & Staehle, 1984, p. 2-1): "Spacecraft Autonomy: The independence of the man/machine flight system from direct, real-time control by the ground over a specified period of time". In this paper, by independence 'withdrawal from or dismissal of control' is meant. The paper is intended to specify the meaning of independence through defining autonomy.

[2] Although this quote is taken from a JPL paper (Easter & Staehle, 1984) it also reflects the SGASAA viewpoint rather well.

Basically, the SGASAA concept proposes that the spacecraft should posses a copy of high-level ground-control command sequences or goals (contained in the Master Schedule), so that in the event of a communication failure the spacecraft is able to plan in order to reach the high-level goals. *Independent planning* is done under supervision of an Onboard Management System that, according to the concept, is able to reschedule the Master Schedule, monitors task execution, co-ordinates and controls the various subsystems and payload managers. Planning is hierarchical in the sense that there is a network of plans with at the top the most abstract Long-Term Operations Plan that defines the objectives for an entire mission, and at the bottom Elementary Commands. Also fault diagnosis and recovery should be performed onboard in case a failure coincides with communication loss. There are three modes of operation, viz. [1] routine mode, in which nominal and expected tasks are executed, [2] crisis mode that deals with unexpected events that results in plan failure and [3] check-out mode that checks the proper functioning of the soft- and hardware.

SGASAA has two major drawbacks that raise questions about the alleged autonomy of a spacecraft with SGASAA functionality, viz. [1] the origin of the Master Schedule and [2] the ability of fault diagnosis and recovery. The Master schedule is completely synthesized at the ground station and it consists purely of expandable macro's. The spacecraft only has the abilities to expand the macro's and set parameters, which cannot be called planning. Moreover, ground control can bypass the Onboard Management System and directly command the payloads which would nullify all possible advantages in the case of independent planning, e.g. the adequacy of decisions based on richer knowledge of the actual situation. Concerning the second mentioned drawback, it was known from the outset that only *expected* failures could be catered for. However, failure recovery should, of course, go *beyond* expected failures.

In spite of the SGASAA aims, the spacecraft remains dependent on the ground station for almost all of its directing functions. In the remaining part of this paper we will develop an alternative concept of autonomy that has a firm theoretical basis and opens up the way to total independent functioning of the spacecraft. We will begin by examining two possible design principles for interacting intelligent systems stemming from Distributed Artificial Intelligence (DAI), viz. Distributed Problem Solving (DPS) and Multi-Agent Systems (MAS).

3 DPS and MAS as Design Principles for Interacting Agents

Distributed AI (DAI) is the field in which systems are designed that have intelligence distributed over a number of distributed nodes or agents. The intelligence consists of knowledge about the problem space (that may or may not be fully specified) and knowledge about problem solving. Applying DAI techniques is useful when the problem under consideration is intrinsically distributed, e.g. geographically when monitoring the movements of vehicles and hypothesizing about their paths, or coordinating the flight movements of aircraft, cf. (Durfee et al., 1987; Durfee et al., 1988; Steeb et al., 1988). DAI systems can generally be designed from two perspectives: Distributed Problem Solving (DPS) or Multi-Agent Systems (MAS). Both systems consist of agents and their organization but DPS takes as its starting point a particular problem with an adequate organization of distributed nodes, while MAS begins with design specifications of individual agents. (Bond & Gasser, 1988, p. 3) define the two fields as follows:

- **DPS** considers how the work of solving a particular problem can be divided among a number of modules, or "nodes", that cooperate at the level of dividing and sharing knowledge about the problem and about the developing solution.
- **MAS** is concerned with coordinating intelligent behaviour among a collection of (possibly pre-existing) autonomous intelligent "agents", how they can coordinate their knowledge, goals, skills, and plans jointly to take action or to solve problems.

However, these descriptions, however, don't supply a distinctive criterion for the two fields since there can be many variations of designs that are intermediate. A reason to qualify a system as either DPS or MAS would, in this view, be only a particular stance with which the system is designed. A *top down* design, taking an organizational perspective, would qualify th system as DPS while a *bottom up* design, aimed at designing individual agents, would render a MAS. Figure 1 depicts this view.

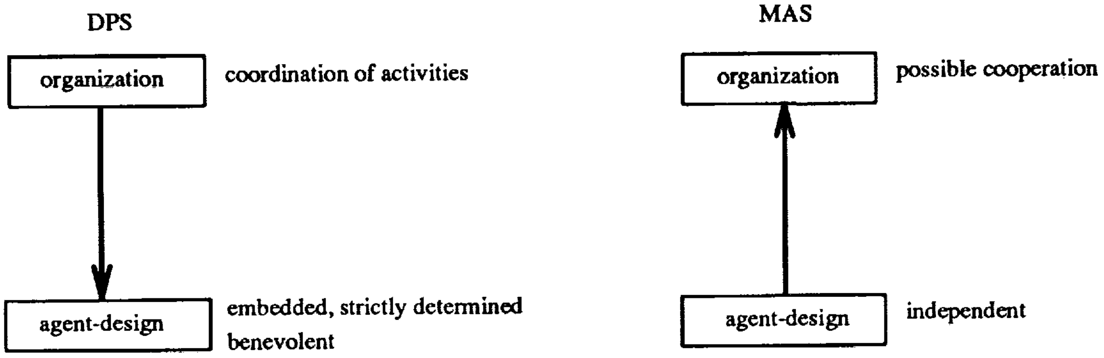

Figure 1: *DPS as top-down and MAS as bottom-up designs. The arrows represent typical consequences following from starting-point design restrictions.*

(Durfee & Rosenschein, 1994) propose individually different utility functions and means to maximize individual payoff as the fundamental difference between DPS and MAS. This is an underspecification, however, because agents can be designed in such a way that utility maximization of the individuals contributes to the higher order, organizational goal. Durfee and Rosenschein therefore consider MAS and DPS properties in more detail to be more specific about the *self-interestedness* criterion for MAS agents.

Another way to look at the difference is by considering DPS as a subset of MAS through a certain number of extra assumptions that hold for DPS, viz. [1] the *benevolence* assumption, i.e. agents are willing to help each other whenever possible, [2] the *common goal* assumption, i.e. DPS agents all have the same goal which may be explicitly represented but may also be embedded in the organization and possible roles agents can assume under certain circumstances, [3] the *centralized designer* assumption, i.e. the designer controls all aspects of agent behaviour in a fully specified environment. To summarize: in DPS the agent design is completely dependent upon the higher order goal and behaviour is completely determined by organizational choices meant to solve a particular problem.

There are a few problems with these criteria. First, as Durfee & Rosenschein observe, even with the mentioned assumptions, DPS systems do not necessarily function optimal due to non-optimal local decisions made by the individual nodes. Also, the extent to which the goals should differ to categorize them as MAS agents is unclear. Finally, it is possible to equip agents with a payoff function that instantiates a high-order goal which makes it difficult to decide whether they are MAS or DPS agents. In fact, this was the reason to view DPS and MAS designs as spanning a continuum in which both designs may have different starting points. Table 1 contains a summary of the contrasting properties of DPS and MAS.

Table 1: *Contrasting DPS and MAS properties*

DPS	MAS
agent design depends on organizational choices	pre-existing / pre-formed agents
benevolent	self-interested
common goals/utility function	individual goals/utility function
fully specified environment	unspecified / partly specified environment
organizational coordination of results and / or tasks	societal cooperation on basis of joint plan formation
global success criteria	situation assessment based on goal states
domain-specific problem solving	individual problem solving / self-maintenance

Hence, neither the individual utility function, nor the restrictions on MAS that define DPS, provide a decisive criterion. The problem is that although the possession of individual utility functions seems a good candidate, it is difficult to determine whether they are dependent on the social goal. In fact, goals in artificial agents are *always* dependent on the goals of the *designer*[3].

Thus, we argue that the only way to be certain that the agents are *self-interested* and that their goal structure doesn't depend on the designer's, is that the agent's goal structure has evolved from scratch in a real environment. This is the case only for living organisms. Hence, *autonomous agents cannot be designed* but rather have evolved by themselves. Still, to be practical, we can maintain a notion of *quasi-autonomy*. An agent is quasi-autonomous if it mimics the functionality of autonomous agents. Autonomous agents have to preserve themselves, which is the first, and necessary, requirement for self-interestedness. Self-preservation, in turn, requires adaptability.

[3] We can be quite fussy about this point and argue that even if agents have random goals, they are still dependent on the designer's goals, because he or she has had reasons (motives) to design the agents' goal structure in that particular way.

It will be clear by now, that SGASAA functionality doesn't come near quasi-autonomous functionality. SGASAA is much closer to the DPS pole, because of the limitless way ground control can influence the goals of the spacecraft. Nevertheless, taking the fact into account that spacecraft indeed face uncertain circumstances and that they should preserve themselves (which is one of the motivations for SGASAA), its design actually *should* be closer to our concept of (quasi-)autonomy. In the next section we will again consider the origin of the goals of adaptive systems and further examine adaptive functionality from a theoretical perspective to specify what autonomous agents should be capable of.

4 Autonomy as Resulting from Constraints on Adaptation

In this section we will give an explanation of the autonomy feature of independency that in our view arises from the property of adaptivity. The first observation, drawn from traditional systems theory, will be that maximally adaptive systems are goal-directed, i.e. they try to lift the discrepancy between measurements from situations and an internally represented goal-state using feedback to guide action. Secondly we will address the origin of the goals in adaptive systems. Since certain goals can't be changed agent-internally, we also have to look at the goal development from a intra-specimen perspective, which will be called the *fylogenesis* of goals.

Finally, we will look at a possible architecture that takes elementary adaptive units as its building blocks to comprise a more general adaptive system that besides the already facilitated functionality of goal achievement, which we shall call *satisfaction behaviour*, also displays goal patching or *avoidance behaviour* when the threat that a particular goal will not be achieved becomes too big.

In this section, we primarily establish that adaptive systems contain a core of stable goals, i.e. a core that can't be influenced. In the next section we will look at what that means for a system that is capable of generating goals *endogenously* or capable of receiving external goals. Figure 2 summarizes what issues will be considered in what order in this and the next section.

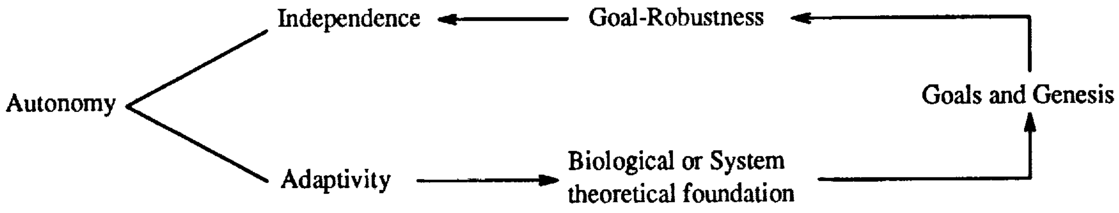

Figure 2: *The two parts of autonomy and their relationship. Independence is justified by the MAS property of adaptiveness to uncertain circumstances.*

4.1 First Ramifications of Adaptivity: a Systems Theoretic Approach

Traditionally, system theory, cf. (Glisson, 1985), roughly divides (linearized) systems into three categories, viz. [1] I/O systems [2] systems with a state representation or a state vector [3] feedback systems. I/O systems doesn't have a 'memory' and their output is a (linear) transformation of the input. Systems with a state representation have an output function which is dependent on the previous state and possibly a direct component from the input. In feedback systems, output is directed back through a function that contrasts output with desired output. A block-diagram of an output-feedback system can be found in figure 3, cf. (Owens, 1978).

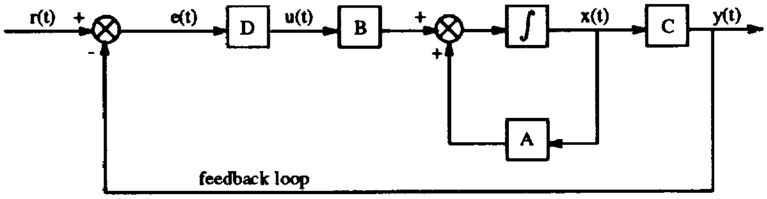

Figure 3: *A constant output feedback-control system.*

The system equations are as follows:

$$\begin{array}{ll} \dot{x}(t) = Ax(t) + Bu(t), & x(t) \in R^n \\ y(t) = Cx(t), & y(t) \in R^m,\ u(t) \in R^l \end{array}$$

The state vector is denoted by x, together with corresponding transformation matrix A it comprises a 'memory'; u is the input vector with transformation B; y the output vector that depends on the state through transformation C. A $m \times 1$ vector of demanded outputs, $r(t)$ can be constructed, to result in an error vector $e(t) = r(t) - y(t)$ that is fed back into the system, resulting in the input $u(t) = De(t)$ where D is a constant $l \times m$ matrix.

An output feedback control (OFC) system is a straightforward extension of the two simpler systems. In addition to the I/O model, 'memory' is added which makes iterated action possible, and in addition to the state-vector system feedback is added, which makes it possible to compare output with desired output.

Comparing output with demanded output is a test for performance, or a test to what extent the situation converges towards the desired state. Demanded output can thus be seen as a *goal* and feedback gives an indication of how closely the system has neared the goal-state[4]. The limitations of the OFC system are that only information is processed that a *priori* was established as goal-relevant.

According to a classification of (Cariani, 1991, p. 786), there are two types of systems below

[4] This system-theoretic classification can very well be mapped on the classification of (Genesereth & Nilsson, 1987). They distinguish [1] tropistic agents (I/O systems) [2] hysteretic agents (state vector systems) and [3] knowledge-level systems (OFC systems).

the adaptive one that don't dispose of the capacity to learn or use a feedback design, viz. [1] fixed computational and [2] fixed-robotic. They do have the capacity to respectively execute pre-specified rules and execute fixed percept-action combinations but not to optimize their percept-action coordination. He calls the OFC system the adaptive device type and there is another more general adaptive system, viz. the *general evolutionary* device type. Cariani's classification can be found in table 2. In the next subsection we will focus on the difference between the adaptive and the general evolutionary system in order to establish the maximally feasible adaptive system and its properties.

Table 2: *Device types according to Cariani, a representative of A-Life*

Device Type	Capacities	Limitations
fixed computational	reliable execution of pre-specified rules	limited to pre-specified rules and states
fixed robotic	reliable execution of fixed percept-action combinations	no feedback or learning from the environment
adaptive	performance-dependent optimalization of percept-action coordination	limited to percept & action categories fixed by the sensors & effectors
general evolutionary	creation of new percept & action categories: performance-dependent optimalization within these categories	time to construct & test new sensors & effectors may be very long

4.2 Passing Fitness Criteria through the Genome

As we have noted in section 3 the genesis of the goals in adaptive systems plays a key rôle in the notion of autonomy. To be genuinely autonomous, the goals of an agent should originate from the objective of self-preservance in an uncertain environment. The issue we will now consider concerns the process of genesis, i.e. how goals can evolve and especially the question whether individual agents are capable of changing or generating *all* of their goals themselves (i.e. *intra-specimen*) or that change happens *inter-specimen*.

According to Cariani there exists an adaptive device that is more general than an OFC, viz. the evolutionary device type. This device is capable of the development of new sensors, establishing new computations on new sensory primitives (what he calls *epistemically autonomous*, we will not consider this property further) and constructing their own performance-measuring apparatuses (this is what he calls *motivationally autonomous*, Ibid. p. 789). Motivationally autonomous agents change their evaluative criteria (what we have called norms or goals) *themselves*.

OFC systems are directed towards the norm, they evaluate output using the norm. If they would be able to change their norm, there would be no guiding criterion because the norm itself has that function. This means that systems capable of changing the norm must do so *arbitrarily*. For systems that must maintain themselves changing the goal that they want to achieve *arbitrarily* involves a high risk. Goals that are generated *arbitrarily* may direct the system to self-destruction. Without a stable prior goal, there is no way new goals can be tested on adversity or beneficiality. It could even be argued that the general evolutionary device type can't be classified as an adaptive system because there is no demanded output, after all, if something is demanded then a random mutation of it is not necessarily demanded.

If we look at the evolution of natural agents, experiments that randomly change the architecture of organisms take place but *inter-specimen* rather than *intra-specimen*. Mutations take place from one generation on to the next and the success of this experiment is determined by fitness criteria[5]. If a specimen matches the fitness criteria well enough, the changes are passed through the genome and remain stable in the next specimen. Comparing possible architectures in terms of systems theory leads to the conclusion that OFC systems have an advantage because they are goal-directed but only if their norm is a proper representation of environmental survival conditions, i.e. if the norm has developed under evolutionary conditions. We call the evolutionary development of goals the *fylogenesis* of the goals.

Changing evaluative criteria *arbitrarily* is unpermittable for individual specimens because it leaves them without any success criteria of their action which would nullify the advantages of feedback. From this we conclude that a more general adaptive device through flexibility of the totality of goals, is not feasible. However, there is an extension of the adaptive device that consists of layers of OFC systems and has important, indispensable functionality. We will turn to this now.

4.3 Reflectiveness in Task Networks of OFC systems

Although Cariani's evolutionary device type doesn't provide a more general adaptive system than the OFC, the adaptive functionality of an OFC *can* be extended. Briefly this can be achieved by recursively linking the OFC systems into a *task network* so that there is a hierarchy with at the top level OFC systems representing the fylogenetic goals (we will call these *primitive goals*) and at the bottom level OFC systems that perform action primitives and intermediate subgoals[6].

In comparison to classical planners, cf. (Charniak & McDermott, 1985, ch. 9), task networks consisting of OFC systems have an important advantage over traditional planning operators. In traditional planning theory monitoring the execution of a plan is identified as one of the most intricate problems (Ibid. p. 489, 524). The decision when to abandon a subgoal (i.e. assessment of the situation to establish the rate of convergence to the goal state), is of prime importance in plan-execution monitoring. This decision can be based on the rate of goal-state convergence. However, a single OFC system is not able to change its action pattern; in fact, it can only *amplify* its attempts. Hence, if convergence is too slow or if the situation diverges from the goal-state, there should be a possibility to change the action pattern altogether.

The creation of a task network with OFC systems is straightforward because each OFC has

[5] Fitness is defined in (Koza, 1992, p. 94) as the probability that an individual survives to the age of reproduction and reproduces. In Artificial Life there are usually other methods of measuring fitness than reproduction, e.g. in a population of artificial ants it can be the number of food parcels eaten.

[6] We will not elaborate much on the linking mechanism because of space limitations. An elaborate discussion of this can be found in (Nilsson, 1994).

a description of its goal state in r. If π is the overall goal and there exists a schedule of $\{r_1 \cup r_2 \cup \ldots \cup r_n\} \supseteq \pi$, a task network for π exists. Execution of any current subgoal can be monitored by examining if the output converges to its desired state:
$(r_p - y_p(t)) - (r_p - y_p(t+1)) \leq c$. Using an abstract matching operator M, then if there is an alternative goal r_q such that $r_q M r_p$ and $x_p(t) \neq x_q(t)$, then an alternative task network can be reassembled on the failure of a subgoal. We will call this the 'reflective' property of the system.

In reflective systems two modes of behaviour can be distinguished: *satisfaction behaviour* in which a task network is synthesized after which all the subgoals are attempted to be satisfied and *avoidance behaviour* in which replanning is initiated to patch up failing (sub)goals. Sole OFC systems are not able to find alternatives for their attempts to satisfy their goals, while reflective systems are (in principle) able to apply alternatives.

With the property of reflectiveness we have completed our specifications for a system that has maximal adaptivity. Before we continue to examine its properties, we will make one last remark about the epistemological status of the ontogenetic goals. In the previous section we argued that a system that has no fixed evaluation criteria is undirected and that the criteria are fixed by environmental constraints passed on through the genome. We now have a network of OFC systems that interact through their goal-state descriptions (the respective r's). Because the OFC systems are hierarchically organized, higher level goals can act as supervisors of lower level goals and change their goal-state depending on performance which creates possibilities for percept-action coordination that Cariani only granted the general evolutionary device type[7].

5 Goal-Robustness, Independence and Autonomy

In the previous section the two main functional modes of goal-directedness, i.e. satisfaction and avoidance behaviour were discussed, and the canonical position of the goals was established through examining the adaptive design. The canonicality of the goals has another implication, viz. the property of goal-robustness. If a system has the property of goal-robustness all of its goals are instantiations of a set of irreducible or *primitive* goals of which the existence was shown in the previous section. The issue in this section is the functionality of the system that can generate *endogenous* or receive *external* goals.

Section 5.1 demonstrates the connection between the autonomous feature of independence as proposed by (Castelfranchi, 1994) and the property of goal-robustness. Independence derives from a mechanism that implements goal-robustness, i.e. only those goals are assimilated and scheduled that are consistent with the primitive goals. Finally, the criteria

[7] His claim that a system is only truly emergent if a system places itself outside the observational frame of the designer is, according to our opinion, mistaken. (Rosen, 1986) has made a case for the informational equivalency of behavioural determining factors, i.c. genome and state, that shows the fundamental impossibility of reducing observations to one of these factors as an explanation. In fact, this is a methodological problem of underdeterminedness of observations through which it is *in principle* not possible to make a distinction between the adaptive - and the general evolutionary device type.

for assimilating directives from other agents are formalized which completes the features necessary for autonomy. Figure 4 schematically shows the canonical position of the goals.

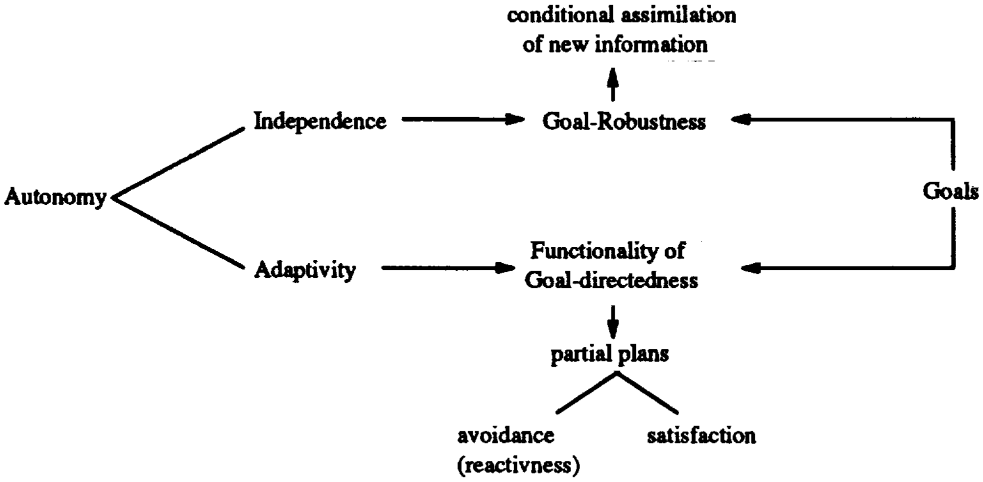

Figure 4: *The two parts of autonomy and their relation. Independence is justified by the MAS property of adaptiveness to uncertain circumstances.*

5.1 Criteria for Autonomy

According to Castelfranchi there are two kinds of autonomy, viz. [1] autonomy from physical context (from environment) and [2] autonomy from social context (from other agents). He refers to [1] as the "Descartes Problem": "Which Agent Architecture guaranties that the Agent is neither completely determined by stimuli (stimulus dependent), nor completely unreactive to environmental changes?" (p. 52).

We interpret [1] as the question of how action can originate from the agent, i.e. endogenously rather than completely from the current situation. The embedded feedback systems architecture, discussed in the previous sections, provides a solution. All the goals in the planning system are instantiations of the fylogenetic goals, and therefore provide an explanation of behaviour that doesn't originate from the current situation. Thus, the task network combines situational appropriateness and conformity to the fylogenetic goals[8]. The execution of the task network is constantly being monitored, adjusting when necessary and even abandoning the current action pattern if it appears to be inadequate. Execution monitoring and adjustment provides the required reactivity. With the ability to pursue endogenous goals, an agent could be said to be *independent* from its environment. Actually, the property that is intuitively closer to independence is *independence from other agents*, which is Castelfranchi's second criterion.

Autonomy from social context means that an agent will not unconditionally follow goals others propose to it. This follows naturally from the fact that primitive goals are stable and

[8] Explanation of behaviour of autonomous agents is *diachronical* rather than *synchronical* as it is in I/O systems.

that the agent's goal-state representations should be consistent, i.e. not have contradictory goals, we will call this the property of *goal-robustness*. Providing the agent with goal consistency is a distinct problem that requires various goals to be contrasted along a consistency measure. It is clear that a consistency measure should be a function of the subjected goal and the primitive goals. In the next subsection we will look at what this function might look like.

5.2 Checking New Goals on Consistency

The question of how the goal state of individual agents changes when agents try to influence each other's goals explicitly, i.e. through communication of directives, is addressed by (Werner, 1989). He contends, as we do, that in real life situations complete plans cannot be communicated (Ibid. p. 7) and that goal states can't be changed by others unconditionally (Ibid. p. 17). However, except for a few well-defined cases in which the organizational structure determines the conditions under which new goals are assimilated (so called *roles*), he doesn't define a function that tests received goals on compatibility with agent-dependent utility functions. We will make a first attempt in order to specify the property of goal-robustness that was introduced in the previous section.

Task networks are usually assembled by searching for task operators that reduce the difference between the current - and the goal state (cf (Charniak & McDermott, 1985, ch. 9)). The result is a *goal conjunction* π that can be matched against the current situation represented in I. In π there may be a number of variables, either to be bound to other operators or to primitives in the situation representation, we will denote this as follows: $\pi\theta$ in which θ represents the set of free variables[9]. On assembly of the plan, the set of differences can be reduced by replacing goal conjuncts by conjuncts that have greater detail and therefore a better match with the situation, until the planner has found a *maximal match substitution*. In effect this is a reduction of the number of free variables to a substitution such that for all other substitutions θ', $|\pi\theta - I| \leq |\pi\theta' - I|$.

In a task network the leaves of the planning tree are matched in this way to the situations. However, we do not only want to know how well a task network fits a particular situation but also how well a particular taks network instantiates the primitive goals, which is ultimately where the agent is directed at. Any subgoal can be tested on fit with a higher-level goal from which a utility value is produced which is maximal when the match is perfect, i.e. when there are no free variables. On perfect match, a task network will completely be executable and it will realize a high-level goal. Analogous to the matching of a task network to a particular situation, a particular instantiation $\pi'\theta$ can be matched to a higher-level goal π. Analogous to maximal fit, maximal utility with respect to goal π is defined as follows:

$$\forall\theta' \, \exists\theta \; \{|\pi'\theta - \pi| \; \leq \; |\pi'\theta' - \pi|\} \tag{1}$$

For a perfect plan the following holds: $\pi'\theta \subseteq \pi \subseteq I$. Overall utility of the instantiation is

[9] Our notation is a mixture of Charniak & McDermotts, of Werners and our own.

defined as follows:

$$E(\pi, \pi', \theta) = \frac{1}{|\pi'\theta - \pi|} \cdot \lambda_\pi \tag{2}$$

In which λ_π is an overall utility value of π or a constant if π is a direct instantiation of a primitive goal. Hence there exists a set of fixed utilities, Φ that is a subset of the total set of utilities and which is indexed to the set of primitive goals Ψ: $\Psi \rightarrow \Phi$; $\Psi \subseteq \Pi$ and $\Phi \subseteq \Lambda$.

The evaluation function enables the planner to decide which goal and instantiation to choose. The utility of a particular goal depends recursively on the match with a primitive goal, hence goals that instantiate a highly rated primitive goal well are preferred above goals that are either poorer instantiations or linked to lower rated primitive goals. In an environment where the planning agent is liable to influence, the evaluation function provides a strong criterion for assimilation of a communicated goal. There are two possible situations: either the received goal is an instantiation of a priorly uninstantiated goal, or it is a replacement of an already existing goal. In the first case the criterion for assimilation is that total utility must increase: $\Sigma_n\ E(\pi_i, \pi'_i, \theta) > \Sigma_n\ E(\pi_i, (\pi'_i + \pi'_q), \theta)$ in which n equals the total number of goals. In the second case the criterion is that if the received goal can be instantiated so that it has a higher evaluation value than an already present goal, it will be assimilated into a task network and executed, otherwise it will be rejected. Formally, the criterion for accepting the new goal π'_2 at the cost of goal π'_1 is: $E(\pi, \pi'_1, \theta) \geq E(\pi, \pi'_2, \theta)$.

6 Conclusions

In the last section we have examined the property of goal-robustness as the last of the design specifications of an autonomous agent. This property has far-stretching consequences for spacecraft control. It means that ground control commands will not be unconditionally accepted, i.e. commands may be rejected when they don't meet the criteria specified above. The relation ground-control/spacecraft becomes one of cooperation in which joint plan formation is possible by exchanging high-level goals and information. This is the way in which independent agents cooperate in a real-life situation (Werner, 1989, p. 7). This organizational structure is more appropriately classified as a MAS. Primarily this is the case because, due to communication limitation, the spacecraft can't 'think' on the ground and it has to take decisions directly in response to its environment (when the situation requires prompt action) or in absence of a consultant (when communication with ground control fails). In this light the expressed position "a fully autonomous space[craft] is neither achievable nor necessarily desirable." (Easter & Staehle, 1984, p. 5-2) has to be revised, while the question "... how long a space platform can perform a given function, even in the presence of new and existing faults, without intervention or direction from ground personnel or equipment" (Ibid.) can be answered by: indefinite, but more likely *better* if ground control *recommendation* is available. In that sense, as we have shown in this paper, we fully agree with the following two standpoints (Ibid. p. 3-4): "No autonomous system is actually free of human supervision; autonomous systems do not replace humans in this sense." because the high-level goals are always originating from humans and high-level goals can't be replaced by the system itself; "[autonomous systems] provide much more flexibility for determining the optimal degree, nature and location of human participation in space activities", indeed they do, because *they* determine the appropriateness of human advice and direction.

References

Aarup, M., Munch, K. H., Fuchs, J., Hartmann, R., & Baud, T. (1994). Distributed intelligence for ground/space systems. In *Proceedings of the 3rd International Symposium on Artificial Intelligence Robotics, and Automation for Space* Pasadena, California, USA: JPL Publication 94-23.

Bond, A. H. & Gasser, L., Eds. (1988). *Readings in Distributed Artificial Intelligence*, San Mateo, California. Morgan Kaufmann Publishers.

Cariani, P. (1991). Emergence and artificial life. In C. G. Langton, C. Taylor, J. D. Farmer, & S. Rasmussen (Eds.), *Proceedings of the second workshop on artificial life*: Addison-Wesley.

Castelfranchi, C. (1994). Guaranties for autonomy in cognitive agent architecture. In M. J. Wooldridge & N. R. Jennings (Eds.), *Proceedings of the 1994 Workshop on Agent Theories, Architectures and Languages* Chichester, West Sussex: John Wiley & Sons.

Charniak, E. & McDermott, D. (1985). *Introduction to Artificial Intelligence.* Addison Wesley.

Durfee, E. H., Lesser, V. R., & Corkill, D. D. (1987). Cooperation through communication in a distributed problem solving network. In M. N. Huhns (Ed.), *Distributed Artificial Intelligence* (pp. 29–58). San Mateo, California: Pitman Publishing\Morgan Kaufmann.

Durfee, E. H., Lesser, V. R., & Corkill, D. D. (1988). Coherent cooperation among communicating problem solvers. In (Bond & Gasser, 1988), (pp. 268–284).

Durfee, E. H. & Rosenschein, J. S. (1994). Distributed problem solving and multi-agent systems: Comparisons and examples. In M. Klein & K. Sharma (Eds.), *Proceedings of the 13th International Distributed Artificial Intelligence Workshop* (pp. 94–104).

Easter, R. W. & Staehle, R. L. (1984). *Space Platforms and Autonomy.* Technical Report JPL D-1973, Jet Propulsion Laboratory, Technology and Space Program Development, NASA/California Institute of Technology, Pasadena, California.

Genesereth, M. R. & Nilsson, N. J. (1987). *Logical Foundations of Artificial Intelligence*, chapter 13, (pp. 307–327). Morgan Kaufmann Publishers: Los Altos, California 94022.

Glisson, T. H. (1985). *Introduction to System Analysis.* New York: McGraw-Hill.

Grant, T. J. (1994). Space/ground systems as cooperating agents. In J. Rash (Ed.), *Proceedings of the 9th Annual Goddard Conference on Space Applications of Artificial Intelligence* (pp. 357–368). Also to appear in: Telematics & Informatics.

Koza, J. R. (1992). *Genetic Programming; On the Programming of Computers by Means of Natural Selection.* The MIT Press.

Nilsson, N. J. (1994). Teleo-reactive programs for agent control. *Journal of Artificial Intelligence Research*, 1, 139–158.

Owens, D. H. (1978). *Feedback and Multivariable Systems.* Southgate House, Stevenage, Herts, England: Peter Peregrinus ltd.

Pidgeon, A. N. B., Seaton, G., Howard, G., & Peters, K.-U. (1992). *Spacecraft Autonomy Concept Validation by Simulation.* Phase 2 Final Report CR(P) 3604, Issue 1, European Space Agency.

Rosen, R. (1986). On information and complexity. In *Complexity, Language and Life: Mathematical Approaches.* Springer Verlag.

Steeb, R., Cammarata, S., Hayes-Roth, F. A., Thorndyke, P. W., & Wesson, R. B. (1988). Distributed intelligence for air fleet control. In (Bond & Gasser, 1988), (pp. 90–101).

Werner, E. (1989). Cooperating agents: A unified theory of communication and social structure. In *Distributed Artificial Intelligence*, volume 2 (pp. 3–36). San Mateo, California: Morgan Kaufmann.