Resources as basic concepts for authority sharing

Stéphane Mercier\textsuperscript{1,2}, Frédéric Dehais\textsuperscript{2}, Charles Lesire\textsuperscript{1}, and Catherine Tessier\textsuperscript{1}

\textsuperscript{1} Onera-DCSD, Toulouse, France
\{firstname.surname\}@onera.fr
\textsuperscript{2} ISAE-Supaero, Toulouse, France
dehais@isae.fr

Abstract. In the context of supervisory control of one or several artificial agents by a human operator, the definition of the autonomy of an agent remains a major challenge. In the case of uninhabited vehicles, errors are not allowed while performance must be as high as possible. Therefore a trade-off must be found between manual control, usually ensuring good confidence in the system but putting a high workload on the operator, and full autonomy of the agents, often leading to less reliability in uncertain environments and lower performance. Having an operator in the decision loop does not always grant maximal performance and safety anyway, as human beings are fallible. Additionally when an agent and a human decide and act simultaneously using the same resources, conflicts are likely to occur and coordination between these heterogenous entities is mandatory. We present the basic concepts of an approach that will aim at dynamically adjusting the autonomy of an agent in a mission relatively to its operator, based on a formal modelling of mission ingredients.

1 Context of the Study, Assumptions and Objectives

In this paper we focus on the autonomy of artificial agents (e.g. uninhabited vehicles, autopilots…) supervised by a human operator and achieving objectives for a given mission. Such agents evolve in a dynamic environment and face unexpected events. Consequently real-time reactions to these events in order to avoid dangerous situations and the loss of the agents themselves are compulsory. Additionally we consider systems where most of operational tasks can be associated with procedures, i.e. tasks must be executed in a precise order and respect strict constraints (as it is the case in aeronautics).

In an ideal context the agents would be able to achieve the mission completely independently from the operator, a case that is hardly likely to occur in reality. However this is a necessary ability for the agents as communication breakdowns between the agents and the operator may occur during the mission. Beyond this extreme case the agents may request the operator’s help anytime for any task when a problem arises. On the other hand the operator her/himself must be free to intervene at any stage of the mission in order to adjust the agents’behaviours according to her/his preferences but also to correct their possible mistakes or improve their performance.
One of the main challenges is conflicts. The human operator’s inputs may interfere with the agents’ plans and break their consistency anytime, even if the inputs are intended to improve a task or to correct an agent’s mistake. As an agent and the operator may both execute actions on their own, it is of great importance that they should remain coordinated so that they should not use the same resources at the same time for different purposes. For example if the autopilot of a UAV and the operator simultaneously “decide” to move the vehicle in different directions, inconsistencies are very likely to appear in the flight and lead to an accident. Therefore conflicts must be detected and solved as soon as possible.

Finally our main objective can be summarized in the following question: why, when and how should an agent take the initiative? When the environment has changed and the agent’s plan needs to be updated? When the operator’s inputs are inconsistent with the procedures and with security constraints? Or when they create conflicts with the current goals?

2 State of the art

While there is no universal definition of autonomy, this concept can be seen as a relational notion between entities about an object [1, 2]: for instance, a subject \(X\) is autonomous with respect to the entity \(Z\) about the goal \(G\). In a social context entities like other agents or institutions may influence a given agent thus affecting its decision-making freedom and its behaviour [3].

In the context of a physical agent evolving in the real world (i.e. an uninhabited vehicle) under the control of a human operator, autonomy can be seen as the ability of the agent to minimize the need of human supervision and to act alone [4]: the primary focus is then rather the operational aspect of the autonomy than the social one. In this situation pure autonomy is just a particular case of the agent - operator relationship, precisely consisting in not using this relationship.

However in practice, as automation within complex missions is not perfectly reliable and is usually not designed to reach the required objectives alone, human supervision is still needed. Moreover it seems that human intervention significantly improves performance over time compared to a neglected agent [5, 6].

Autonomy levels

[7] first proposed a classification for operational autonomy based on a ten-level scale. This model remains quite abstract as it takes into account neither the environment complexity nor the mission context. However it provides an interesting insight into the interactions between an operator and an agent. This model has later been extended, using the same scale applied to a four-stage cognitive information processing model (perception, analysis, decision-making and action) [8]. Based on the same principles other scales for autonomy classification have also been proposed, e.g [9].

---

3 Uninhabited Air Vehicle
Other approaches aim at evaluating an agent’s autonomy in a given mission context, like MAP [10], ACL [11] or ALFUS [12]. The latter proposes to evaluate autonomy according to three aspects: mission complexity, environmental difficulty and human interface. However this methodology aggregates many heterogeneous metrics and the meaning of the result is hard to evaluate. Moreover a qualitative step is necessary especially to set weights on the different tasks composing a mission and evaluate their importance.

The idea that operational autonomy can be graduated leads to the concept of adjustable autonomy or shared authority. The main principle is that machine and human abilities are complementary and they are likely to provide better performance when joined efficiently than when used separately [13]. A physical agent is thus capable of evolving at several predefined autonomy levels and switches levels according to the context. A level is defined by the complexity of the commands [14] or the ability to perform tasks without the need of operator’s interventions [5]. The major limitation we can see in these approaches is the a priori definition of the levels, the static distribution of tasks among entities at each level and the fact that the number of levels is necessarily limited. Interactions between the agent and the operator are thus restricted to a given set and are determined by autonomy levels, there is no possibility of fine dynamic task sharing.

To add more flexibility, [15] endow agents with learning capabilities based on Markov Decision Processes (MDP) allowing them to better manage the need for human intervention. Agents can define their own autonomy levels from the user’s provided intentions. However this method does not seem to be directly applicable to critical systems as the behaviour of learning agents facing unexpected situations is hard to validate. Moreover the operator’s interactions are restricted to the agent’s needs.

The approach of [16] adds more human control on the agent. Levels are not defined in a static way but come from a norm: permissions and restrictions describing the agent’s behaviours are set by the operator. In order to do so, the operator has to create a complete set of rules like “In case of medical emergency, consult the operator to choose landing location”. The major issues associated with such an approach are the high number of rules to provide and the risk of conflict between rules. Anyway the autonomy of the agent is completely human-supervised and the agent has no possibility to adapt itself.

Sliding autonomy [17] consists in determining whether a task should be executed by the agent alone or by the operator using manual control; there is no direct reference to autonomy levels. Roles are not shared at the mission level but are reconsidered for each action to realize. However it seems that the range of human-agent interactions is really restricted as each task is performed either ”completely autonomously” or ”completely through teleoperation”.

In contrast, collaborative control is an approach aiming at creating dialogs between the operator and the agent [18]: the agent sends requests to the human operator when problems occur so that she/he could provide the needed support. This is again a restriction of all possible interactions: only dialog is used whatever
the circumstances. In practice almost all interactions are initiated by the agent’s requests and the operator acts almost exclusively as a support, she/he has not much initiative.

[19] have studied two authority sharing modes on a simulated space assembly task, SISA (System-Initiative Sliding Autonomy) where only the agent can request the operator’s support and MISA (Mixed-Initiative Sliding Autonomy), where the operator can also intervene anytime. The allocation between the agent and the operator is realized separately for each task according to statistics to determine which entity will be the most efficient, which does not seem sufficient for a critical mission where errors are not allowed. However sharing at the task level is an interesting idea as it provides the most adaptive solution to the mission.

As shown by the literature review it is often interesting to join human and machine abilities to carry out a mission and adjustable autonomy seems a good principle. However the fact that the human operator also is fallible is often neglected. While it seems reasonable that the operator should keep the control over the agent, in most of the studies the operator’s inputs are not evaluated and accepted “as they are” by the agent. Moreover the simultaneous decisions and actions from an artificial agent and a human agent might create misunderstandings and lead to conflicts and dramatic situations [20].

3 First formalization

3.1 Resources

The first step to get a formal and operational definition of adaptive autonomy is to formalize the basic concepts of a mission operated by physical agents and human operators.

Our approach consists in modeling the resources needed for the mission accomplishment and how they may be shared (among the operators, the agents, the environment). Authority sharing during the mission consists in detecting conflicts on the resources, assess their impact on the mission and manage them in order to satisfy the objectives while satisfying some criteria like security or performance.

We will consider a mission as a set of resources that are arranged in time so that the goal of the mission should be reached. Resources include: physical objects, energy, permissions, information, algorithms, logic conditions and tasks. The plan to reach the objectives of the mission is built from the resources and the constraints that connect them with each other.

3.2 Properties of the resources

Resources interact with each other as a result of the allocation process, with a using resource on one side and one or several used resources on the other side. However, interactions affect resources and their internal states in different ways depending on their characteristics. In order to model the different kinds of
resource interactions, we have identified the following properties, starting from the *used resource* perspective:

- **Consumability**: a *consumable resource* is available in a limited quantity; once it is spent, no using resource can use it any more. A *non-consumable* resource is restored in its entirety after use.
- **Shareability**: a *shareable resource* can be allocated to several using resources simultaneously. A *non-shareable resource* can be allocated to only one using resource.
- **Preemptability**: a resource can be *preempted* if, while it is used by one using resource, it can be taken by another one thus rejecting the first. Notice that a resource cannot be shareable and preemptable.

A *using resource* is characterized by its dependency upon the used resource(s), which is one of the following:

- **Initialization-Dependent**: the using resource needs the used resource only to become available; if the used resource disappears, the using resource remains.
- **Presence-Dependent**: the using resource needs the used resource to remain available; if the used resource disappears, the using resource disappears too.

### 3.3 Resource: a generic Petri net model

Figure 1 (top) shows three resources, each one being represented by the same Petri net. This generic representation shows (1) the properties of the resource (see above) and (2) the three possible states of the resource: absent (A), present and not allocated (not used) (B) and allocated (used) (C). The transitions from one state to another are triggered by external events coming from the situation assessment function or other resources, changing the marking of the net according to the properties of the resource.

The generic resource model includes the properties from the used resource perspective. To represent the resource dependencies, interface nets between the used and the using resources are designed (see figure 1 (bottom)).

The interface net can be seen as a tool to match the availability of the used resource (offer) with the request coming from the using resource. This results in the allocation process, when offer and request are matched. It corresponds to a transition fusion between the resource and the interface nets.

Additionally, the interface net shows the status of a request at a given time, whether pending or being satisfied. It represents the dependency property, with the consequences on the using resource. As the dependency relationship of the using resource is specific to each needed resource, there are as many interface nets as resources required by the using resource.

### 3.4 Example

Let us illustrate the concepts defined above on a small example.
We consider three resources that are bound to each other within the mission
plan: the first resource is a GPS, the second one is the position information, and the last one is a navigation task requiring the position information during its execution.

The properties of the resources are the following: GPS is a non-consumable, non-shareable and non-preemptable resource, as it is dedicated to the production of the position information. Position is a non-consumable, shareable and GPS presence-dependent resource: it is produced by the GPS only when it works properly and it can be used by several resources simultaneously. Finally navigation is a consumable, non-shareable, and non-preemptable resource; it is position presence-dependent.

Except the allocation transitions, that fire through transition fusion, all other connections between nets are through message passing: the firing of a transition creates a new event with is passed to the other nets, firing activated transitions waiting for it.

Let us see how transitions are fired in the example, starting from the status presented on Figure 1: GPS available and not allocated, position absent and navigation absent.

- An external request (from the operator for example) triggers the interface of navigation [transition 1];
- this triggers the interface of position, which in turn triggers the interface of GPS [transitions 2-3];
- The fused allocation of GPS resource and interface nets are activated and fired [4-5]: GPS is allocated to position.
- GPS interface net gets to the "satisfying request" state, which triggers the resource net of position[6].
- The fused transitions of position resource and interface nets are activated and fired: position is allocated to navigation [7-8].
- The allocation of position interface net triggers the resource net of navigation [9], which is allocated to the external requester [10-11].
- the state of navigation is "allocated", which corresponds to its execution state.
- When navigation reaches its goal, an event is triggered in navigation interface net by the situation assessment [12], its resource net is triggered [13].
- This triggers the interface net of position [14], which in turn triggers the interface net of GPS [16]: position and GPS are thus released [15-17].
- The release of GPS triggers the resource net of position (destruction), as it is GPS presence-dependent [18].
- If the situation assessment detects that GPS gets faulty (bad reception for example) in GPS resource net [19] before navigation goal is reached, this impacts GPS interface net [20], and in turn position resource net (destruction) [21];
- this also affects position interface net [22], and in turn triggers navigation resource net (destroying it, as it is presence-dependent) [23];
- finally the external requester of navigation would be notified of the rejection of the request [24].
Fig. 1. Example: resources interdependencies
4 Conflicts on resources

4.1 Agent Plan Construction

The generic resource and interface Petri nets make it possible to represent the logical dependence within a unmanned vehicle - operator system, i.e. how the state changes of a resource or identified events may affect other resources. The Petri nets are also basic modules to build the mission plan.

In a dynamic open environment, it is impossible to predict all events that may occur and disrupt the plan execution. For a physical agent, a physical resource (e.g. a sensor, the engine, etc.) may break down or become faulty; and the interactions and orders coming from the human operator cannot be predicted. Instead of building a plan trying to anticipate all possible disrupting events and including the appropriate answers, our approach consists in generating a nominal plan only. If this plan is impaired during execution, a new plan (or partial plan) will be generated, taking into account the new constraints on the resources.

The resource representation (resource and interface nets) is the state representation of resources and the successive markings are chronicles of allocations and releases. As resources are implemented for end-users, i.e. the unmanned vehicle, the operator, the environment, the procedures, the Petri nets also track which entity requested, preempted, released which resource. Consequently, the past, current and future states of the Petri nets can be compared with what was, is and will be expected given the nominal plan. Indeed situation assessment gathers information from sensors, monitors the resource states and can detect inconsistencies between the predicted states expected from the nominal plan execution and the observed states. This leads us to the concept of conflict.

4.2 Plan execution and conflicts

A conflict appears during plan execution when inconsistencies are detected within resources. Situation assessment allows inconsistencies to be detected and the involved resources to be to identified. As the history of markings is available, the using entities are tracked. This makes it possible to classify conflicts in several categories, which are listed in table 1 (vertically the entity responsible for plan disruption, horizontally the entity whose plan is affected).

4.3 Authority Sharing

When a conflict is detected within a subset resources, authority sharing consists in the dynamic reallocation of resources (including task sharing between the operator and unmanned vehicles), depending on the objective of the mission, the procedures, risk assessment, the authority status (e.g. in most cases, the human operator should have control over an artificial agent) and ethical considerations.
Conflict | Operator | Agent | External World | Procedures
--- | --- | --- | --- | ---
Operator | Contradictory simultaneous or successive Orders | Prioritary Order, contradictory to agent’s plan | Physically unworkable Order | Violation (deliberately or not)
Agent | Prioritary Action, contradictory to operator’s actions | Failure | Inconsistency between the agent’s plan and actual measures | Violation (with operator’s authorization or not)
External world | Invalidation of operator’s action | Invalidation of agent’s plan or actions | (external world is supposed to be consistent) | Violation
Procedures | Modification of procedures being executed | Modification of procedures being executed | Inadapted Procedures | Procedure inconsistencies

Table 1. The different conflict categories

(e.g. veto power or mandatory approval / supervision by the human before risky actions are run by an agent). The preemptability property of resources in the plan illustrates how an artificial agent interacts with the operator at the authority level, allowing the human operator to override its control on parts of the plan or not. Eventually this framework will define the agent’s initiative, how it should react to unexpected events, to their anticipated consequences on the plan and how it should interact with the operator, resulting in adaptive autonomy.

5 Experimental Environment and Scenario

In order to test our approach for adaptive autonomy and shared authority with concrete applications, the framework for experiments in real conditions with human operators interacting with “autonomous” vehicles is already being designed.

5.1 The Scenario

The scenario is the localization and assessment of a fire by a UGV\(^4\) in a partially unknown area. The mission for the UGV and the operator consists in looking for starting fires around a factory or a facility and determining its properties (localization, size, dynamics) so that it could be quickly put out. The area is hardly accessible, dangerous and partially unknown (no precise and updated map available). Additionally, the scenario could be extended with the possibility for the UGV to carry an extinguisher. This would allow the UGV to directly put out a very starting fire or delay a fire evolution in a given area, e.g. close to sensitive items. As the extinguisher would be very small, its use would have to be carefully chosen. Figure 2 shows the scenario.

Several operational assumptions are made:

\(^4\) Uninhabited Ground Vehicle
– The area where the UGV evolves is divided into two parts: the start area which is known (a map is available), the search area which is partially unknown;
– the known area includes obstacles to avoid, but there are localized on a map;
– the human operator has no direct visual contact with either the UGV nor the outdoor environment;
– there are sensitive items in the known area, which have to be protected against the fire threat coming from the partially unknown area;
– the fires may evolve, possibly blocking known paths or endangering the UGV;
– a fire evolution is determined by the objects that can burn;
– the access paths to the search area are limited and narrow, making the access to the zone difficult.

Additionally, some hazards may impair the mission:
– communication breakdowns between the UGV and the operator;
– dynamic and uncertain environment in the search area (obstacles, fires);
– possible loss of GPS positioning;
– sensor failures.
5.2 The Experimental Set-up

ISAE\textsuperscript{5} is developing an experimental set-up composed of a ground station and several Emaxx UGVs. The UGVs may be controlled either using a remote control (in case of problems) or a graphical interface (normal use). They carry several sensors (GPS, inertial sensors, scenic camera, ultrasounds, odometry) and are able to follow a set of waypoints autonomously. Algorithms are currently being developed to be implemented onboard in order to equip them with decision abilities (planning, situation assessment, authority sharing management).

A wizard of Oz user interface is also being developed as it offers greater possibilities to control "unexpected" events during the experiments (e.g.: communication breakdowns, sensor failures).

6 Conclusion and Future Work

We have presented the general principles and some basic concepts for an approach of operational adaptive autonomy. Resources, including the agent and operator’s tasks, are the key items to build consistent plan to reach mission objectives, but also to determine consequences of unexpected events on it, whether it is a change in the environment or an operator’s intervention. They represent the ground on which authority sharing considerations can be built. Conflicts can be detected and classified depending on the entities that disrupt the plan. Consequently task reallocation within the system is performed so that conflicts could be solved safely with every entity being aware of what is being done, resulting in adaptive autonomy.

Current work focuses on dynamic plan generation, and on the fine identification of what precisely is involved in authority sharing, what metrics at the resource level can be incorporated to determine the priority of an entity over another one depending on the situation. These concepts have to be operationalized then implemented on a real UGV platform (Emaxx) in order to conduct experiments with human operators\textsuperscript{6}. Reliability, overall performance and the operator’s satisfaction will allow us to assess our concepts for adaptive autonomy in real conditions.

References


\textsuperscript{5} Institut Supérieur de l'Aéronautique et de l'Espace, resulting from the merging of ENSICA and SUPAERO
\textsuperscript{6} see section 5
