Chapter 12

Autonomic Management of Space Missions

Michael G. Hinchey, James L. Rash, Walt Truszkowski, [*]
Christopher A. Rouff, [†] *Roy Sterritt* [‡]

12.1 Introduction

With NASA's renewed commitment to outer space exploration, greater emphasis is being placed on both human and robotic exploration. Even when humans are involved in the exploration, human tending of assets becomes cost-prohibitive or in many cases is simply not feasible. In addition, certain exploration missions will require spacecraft that will be capable of venturing where humans cannot be sent.

Early space missions were operated manually from ground control centers with little or no automated operations. In the mid-1980s, the high costs of satellite operations prompted NASA, and others, to begin automating as many functions as possible.

In our context, a system is autonomous if it can achieve its goals without human intervention. A number of more-or-less automated ground systems exist today, but work continues with the goal being to reduce operations costs to even lower levels. Cost reductions can be achieved in a number of areas. Ground control and spacecraft operations are two such areas where greater autonomy can reduce costs. As a consequence, autonomy is increasingly seen as a critical approach for robotic missions and for some aspects of manned missions.

Although autonomy will be critical for the success of future missions (and indeed will enable certain kinds of science data gathering approaches), missions imbued with autonomy must also exhibit autonomic properties. Exploitation of autonomy alone, without emphasis on autonomic properties, will leave spacecraft vulnerable to the dangerous environments in which they must operate. Without autonomic properties, a spacecraft may

[*]NASA Goddard Space Flight Center, Information Systems Division, Greenbelt, MD, USA
Email: {michael.g.hinchey, james.l.rash, walt.truszkowski}@nasa.gov
[†]SAIC, Advanced Concepts Business Unit, McLean, VA 22102, USA
Email: rouffc@saic.com
[‡]University of Ulster, Northern Ireland
Email: r.sterritt@ulster.ac.uk

be unable to recognize negative environmental effects on its components and subsystems, or may be unable to take any action to ameliorate the effects. The spacecraft, though operating autonomously, may then sustain a degradation of performance of components or subsystems, and consequently may have a reduced potential for achieving mission objectives. In extreme cases, lack of autonomic properties could leave the spacecraft unable to recover from faults. Ensuring that exploration spacecraft have autonomic properties will increase the survivability and therefore the likelihood of success of these missions. In fact, over time, as mission requirements increased demands on spacecraft capabilities and longevity, designers have gradually built more autonomicity into spacecraft. For example, a spacecraft in low-earth orbit may experience an out-of-bounds perturbation of its attitude (orientation) due to increased drag caused by increased atmospheric density at its altitude as a result of a sufficiently large solar flare. If the spacecraft was designed to recognize the excessive attitude perturbation, it could decide to protect itself by going into a "safe-hold mode where its internal configuration and operation are altered to conserve power and its coarse attitude is adjusted to point its solar panels toward the Sun to maximize power generation. This is an example of a simple type of autonomic behavior that has actually occurred. Future mission concepts will be increasingly dependent on space system survivability enabled by more advanced types of autonomic behaviors.

12.2 The Role of Autonomicity in NASA Missions

NASA needs to exploit autonomicity in its future missions in order to ensure that they can operate on their own, without human intervention or guidance, to the maximum extent possible. A case can be made that *all* of NASA's future systems should be autonomic, or indeed that all complex systems should be autonomic [7], and exhibit the four key properties of autonomic systems: being self-configuring, self-optimizing, self-healing, and self-protecting [4, 5, 10].

The following discusses the need for each of these autonomic properties in NASA missions. However, it should be noted that these four properties, and other, emerging, self-* properties are not mutually exclusive.

Self-configuration is needed in NASA missions because the nature of the mission may change as time goes on. New or different science may need to be analyzed based on data collected, or if one science instrument fails or deteriorates, mission goals could change and result in increased use of other onboard instruments instead. Reconfiguring the spacecraft or surface rover may be necessary when batteries or solar cells are deteriorating. In this case, unnecessary instruments or functions may need to be shut down to reduce the electrical load and the remaining systems may need to be reconfigured to take this into account.

Self-optimization is needed because the spacecraft, science instruments, and the science being collected may change as the mission proceeds, and the instruments may need to be adjusted or calibrated. Also, the spacecraft might optimize its operations over time by learning more about the environment in which it operates and the object or phenomenon it is observing. For constellations (several spacecraft flying in formation) or swarms of spacecraft (whereby a large number of craft mimic swarms in nature), vehicles will have to constantly adjust their mutually relative positions due to drift, or optimize themselves when members of the constellation/swarm drop out due to malfunctions or other problems.

Self-healing is needed when a spacecraft is damaged or its software is corrupted, or when a member of a swarm or constellation is lost. An example of software self-healing, representing a capability of many, if not most, missions that have actually flown, occurs when a spacecraft is hit by a large amount of radiation and the software is corrupted. The self-healing occurs when the spacecraft recognizes the damage and then restores the affected code, perhaps from another spacecraft or from mission control. In a swarm or constellation, self-healing could occur when the swarm or constellation moves another

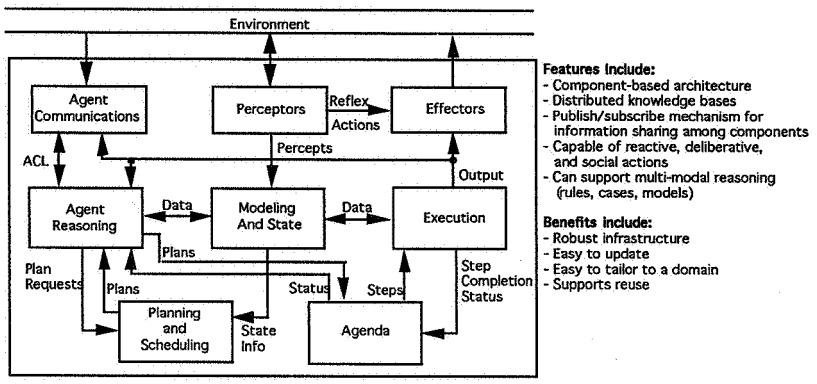

Figure 12.1. *ACT Agent Architecture.*

spacecraft into the place of one that is lost.

Self-protection is needed to keep the spacecraft out of harm's way. For example, solar flares release charged particles that can cause damage to electronics. If a solar flare can be detected, the spacecraft can put itself into sleep mode until the danger has passed. Another example would be a rover on Mars. Dust storms can cause damage to many systems. When a dust storm is sensed, the rover could cover itself or go to a better protected area, such as a rock outcropping or other sheltered area.

12.3 ACT

A major contribution to NASA's goals of achieving greater autonomy has been Goddard Space Flight Center's progress towards using agent technology in operational systems. The Agent Concept Testbed (ACT) [9, 11], is discussed here.

The motivation behind ACT was to develop a more flexible architecture than LOGOS for implementing a wide range of intelligent or reactive agents. After developing ACT, sample agents were built to simulate ground control of a satellite constellation mission as a proof of concept.

Agents in ACT are built using a component architecture, where a component can be easily swapped in and out for easy removal of unneeded components for reactive agents and the inclusion of the necessary components to implement intelligent agents.

ACT also allows for new technologies to be added as they become available without affecting previously implemented components. A simple (reactive) agent can be designed by using a minimum number of components that receive percepts (inputs) from the environment and react according to those percepts. A robust agent may use more complex components that allow the agent to reason in a deliberative, reflexive, and/or social fashion. This robust agent would maintain models of itself, other agents, objects in the environment, and external resources. Figure 12.1 depicts the components involved in a robust ACT agent.

The following are the components of ACT.

Modeler: The modeling component maintains the domain model of an agent, which includes models of the environment, other agents, and the agent itself. The Modeler is also responsible for reasoning with the models to act proactively and reactively with the environment and events that affect the model's state.

Reasoner: The Reasoner component works with information in its local knowledge base as well as model and state information from the Modeler to make decisions and formulate goals for the agent. Currently, the Reasoner works more in a reactive manner.

Planner/Scheduler: The Planner/Scheduler component is responsible for any agent-level planning and scheduling. The planning component receives a goal or set of goals to fulfill in the form of a plan request. This typically comes from the Reasoner component, but may be generated by any component in the system.

Agenda/Executive: The Agenda and the Executive work together to execute the plans developed by the Planner/Scheduler. The Executive executes the steps it receives from the Agenda. If the preconditions are met, the action is executed. When execution finishes, the Executive evaluates the post-conditions, and generates a completion status for that step. The completion status is then returned to the Agenda.

Agent Communications: The agent communication component is responsible for sending and receiving messages to/from other agents. The component takes an agent data object that needs to be transmitted to another agent and converts it to a message format understandable by the receiving agent.

Perceptors/Effectors: The Perceptors are responsible for monitoring the environment for the agent. Any data received by the agent from the environment, other than agent-to-agent messages, enters through Perceptors. The Effector is responsible for effecting or sending output to the agent's environment. Any agent output data, other than agent-to-agent messages, leaves through Effectors.

Agent Framework: The framework provides the base functionality for the components as well as the inter-component communication facility. The framework allows components to be easily added and removed from the agent while providing a standard communications interface and functionality across all components.

12.3.1 Example ACT Scenario

Figure 12.2 illustrates an operational scenario involving a possible ACT agent community for a nanosatellite constellation. It is based on the idea of a ground-based community of proxy agents—each representing a spacecraft in the nanosatellite constellation—which provide for autonomous operations of the constellation. Autonomy can be achieved without the use of agents, and has been achieved in numerous NASA missions, but the example presented here is used to illustrate a solution afforded by an agent framework. Other scenarios for the migration of a community of proxy agents to the spacecraft are discussed in terms of space-based autonomy concepts in [1].

In this scenario there are several nanosatellites in orbit collecting magnetosphere data. The Ground Control Center (GCC) makes contact with selected spacecraft when they come into view according to the schedule. The agents that make up the GCC are:

Mission Manager Agent (MMA) coordinates the agent community in the GCC, manages mission goals, and coordinates with the Contact Manager Agent.

Contact Manager Agent (CMA) coordinates ground station activities, communicates with the spacecraft, and sends and receives data, commands, and telemetry.

User Interface Agent sends data to users for display and gets commands for the spacecraft.

GCC Planner/Scheduler Agent plans and schedules contacts with spacecraft via external Planner/Scheduler.

Spacecraft Proxy Agents keep track of spacecraft status, health and safety, etc. The proxies notify the Mission Manager Agent when anomalies occur that need handling.

Each of the above agents registers with the GCC Manager Agent. The GCC Manager Agent notifies them when a contact is approaching for their spacecraft, whether another agent is going to be added to the community, and how to contact another agent. The following is a spacecraft contact scenario that illustrates how the agents work with the GCC Manager Agent:

- Agents register with the GCC Manager Agent at startup.
- GCC Planner/Scheduler Agent communicates with the Proxy Agents to get spacecraft pass time data. It then creates a contact schedule for all orbiting spacecraft.
- GCC Manager Agent receives the schedule from the GCC Planner/Scheduler Agent and gives details of the next contact to the Contact Manager Agent.
- The Contact Manager Agent contacts the spacecraft at the appropriate time and downloads the telemetry and sends it to the appropriate spacecraft Proxy Agent for processing.
- The spacecraft Proxy Agents process the telemetry data, update the spacecraft's status, evaluate any problems, and send any commands to the Contact Manager to upload.
- If a Proxy Agent determines a problem exists and an extended or extra contact is needed, it sends a message to the GCC Planner/Scheduler Agent, which re-plans the contact schedule and redistributes it to the GCC Manager.
- The Contact Manager downloads data from, and uploads any commands to, the spacecraft as instructed by the spacecraft Proxy Agent.
- The Contact Manager agent ends the contact when scheduled.

An example of a typical contact with a satellite would be:

- The Contact Manager Agent (CMA) receives an acquisition of signal (AOS) from a spacecraft. The GCC is now in contact with the spacecraft.
- The CMA requests the spacecraft to start downloading its telemetry data and sends the data to its proxy agent.
- The proxy agent updates the state of its spacecraft model from the telemetry received. If a problem exists, the Mission Manager Agent is contacted and appropriate action (if any) is planned by the system.
- The Contact Manager Agent analyzes the downloaded telemetry data. If there is a problem, the CMA may alter the current contact schedule to deal with the problem.
- The CMA executes the contact schedule to download data, delete data, or save data for a future contact.
- The Mission Manager Agent ends contact.

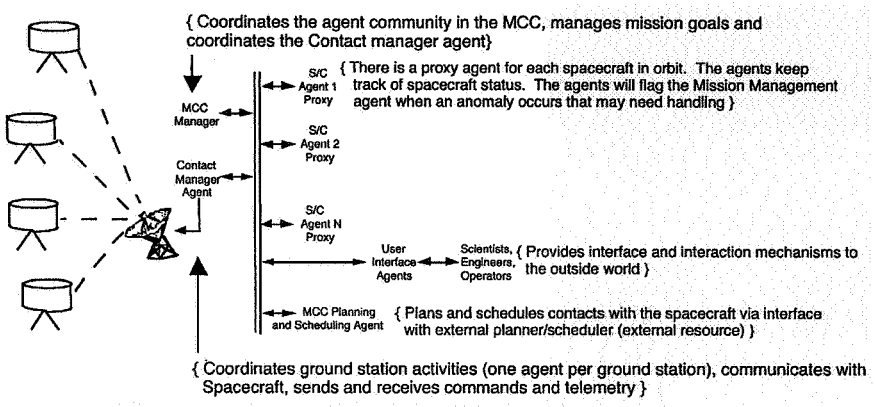

Figure 12.2. *Example ACT Agent Community.*

12.4 A Concept NASA Mission: ANTS

The NASA Autonomous Nano-Technology Swarm (ANTS) mission [1, 2] will be made up of swarms of autonomous pico-class (approximately 1kg) satellites that will explore the asteroid belt.

There will be approximately 1,000 spacecraft involved in the mission, launched from a transport ship. Many of these will be lost due to collisions with each other and with asteroids. The surviving spacecraft will form sub-swarms that will collect science data and return it to earth.

This mission, illustrated in Figure 12.3, will involve a high degree of autonomy, and autonomic properties will enhance its survivability. To implement the mission, a heuristic approach is being considered that uses an insect analogy of hierarchical social structure based on the following spacecraft hierarchy.

Approximately 80 percent of the spacecraft will be workers (or specialists) that will have a single instrument onboard. Some will be coordinators (called leaders) that will have rules that decide the types of asteroids and data the mission is interested in and will coordinate the efforts of the workers. The third type of spacecraft are messengers, which will coordinate communications between the workers, leaders, and Earth. Each worker spacecraft will examine asteroids it encounters and send messages back to a coordinator that evaluates the data and sends other spacecraft to the asteroid if necessary.

Team leaders contain models of the types of science they want to perform. Relevant goals are communicated to the messenger spacecraft that then relay them on to the worker spacecraft. The worker spacecraft then take measurements of asteroids using their specialized instruments until data matches the goal that was sent by the leader. If the data matches the profile of the type of asteroid that is being searched for, an imaging spacecraft will be sent to the asteroid to ascertain the exact location and to create a rough model prior to the arrival of other spacecraft.

12.5 Autonomic Properties of ACT

The various operational scenarios of ACT exhibit at least three types of autonomic functionality and some of a fourth. The autonomic functionalities exhibited are: self-configuring

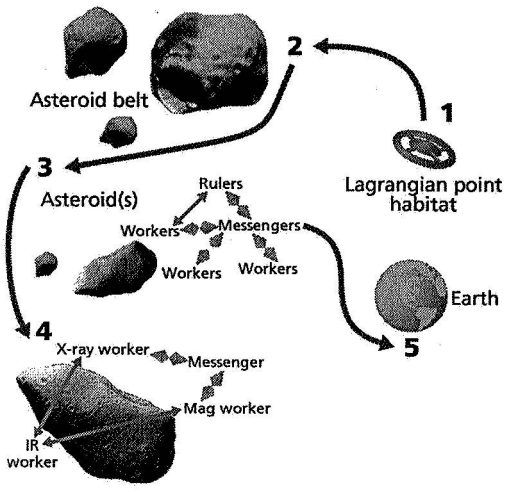

Figure 12.3. *ANTS Concept Mission.*

(adaptation to changing environment), self-optimizing (steps to maximize utilization), self-healing (ability to recover from anomalies), and self-protecting (protect against failures). The following further discusses ACT's autonomic properties.

Self-configuration. As an example of this property, when ACT detects, from analysis of telemetry, that there is a problem, the Contact Manager alters the current satellite contact schedule to enable the problem to be addressed. What is being reconfigured, in this case, is the spacecraft functionality for managing communications contacts with ground systems and controllers.

Self-optimization. As an example of this property, consider what happens when a Proxy Agent determines that a problem exists with its spacecraft. When this situation arises, a replanning/rescheduling activity occurs to optimize the behavior of the entire ACT system.

Self-healing. Consider what happens when a Proxy Agent detects a problem with its associated spacecraft. Following a diagnosis of the problem (which may involve access to the human component of the ACT) corrective actions, in the form of commands, are generated and made ready for transmission to the affected spacecraft. This problem-diagnosis/corrective-action cycle is a major part of ACT's self-healing capability.

It should be noted that the three autonomic responses discussed above all stem from ACT's determination that a problem has occurred. In attending to the problem, ACT reconfigures, tries to optimize its operations, and proceeds to diagnose and solve the identified problem.

Self-protection. ACT is self-protecting in the sense that it constantly monitors the spacecraft systems and modifies its operations if a parameter ranges outside its normal bounds. In addition, it also has self-protection through validation of system commands to insure that command sequences executed will not harm the spacecraft or put it in a position where it could be harmed.

12.6 Autonomic Properties of ANTS

ANTS has an overall requirement to prospect thousands of asteroids per year with large but limited resources. It is anticipated that, to accomplish this, there will be approximately one month of optimal science operations at each asteroid prospected. A full suite of scientific instruments will be deployed at each asteroid. The resources of ANTS will be configured and re-configured to support concurrent operations at hundreds of asteroids over a period of time.

The mission will exhibit high levels of self-management, and in particular exhibits the four self-* properties discussed earlier.

Self-configuration. As asteroids of interest are identified, appropriate teams of spacecraft are configured to realize optimal science operations at the asteroids. When operations are completed, the team disperses for reconfiguration at another asteroid. Reconfiguring may also be required as the result of a failure or anomaly. The loss of a given worker may result in the role of that worker being performed by another. Loss of communication with a worker may mean that the system has to assume loss of the worker, and the role may be allocated to another spacecraft. Loss of use of an instrument by a worker may require the worker to take the role of a communication device.

Self-optimization. Optimization of ANTS is accomplished at the individual level as well as at the system level. These optimizations are:

- Rulers learning about asteroids;
- Messengers adjusting their position;
- Workers learning about asteroids.

Optimization at the ruler level is primarily through learning. Over time, rulers will have collected data on different types of asteroids and will have learned the characteristics of the types of asteroids that are of interest and the types of asteroids that are difficult to orbit or observe. From this information, the system as a whole is being optimized, since time is not being wasted on asteroids that are not of interest.

Optimization for messengers is achieved through positioning. Messengers need to provide communications between the rulers and workers as well as back to Earth. This means that a messenger will have to be constantly adjusting its position to balance the communications between the rulers and workers and adjusting its position to send data to Earth.

Optimization at the worker level is primarily through experience gained with asteroids. As a worker observes asteroids and builds up a knowledge base of the different characteristics of asteroids, a worker may be able to automatically skip over asteroids that are not of interest.

Self-healing. The view of self-healing here is slightly different from that given in [4]. ANTS is self-healing not only in that it can recover from mistakes, but self-healing in that it can recover from failure, including damage from outside force. In the case of ANTS, these are non-malicious sources: events such as collision with an asteroid, or another satellite, loss of connection, etc., will require ANTS to heal itself by replacing one spacecraft with another.

ANTS mission self-healing scenarios span the range from negligible to severe. A negligible self-healing would be where one member of a redundant set of gamma ray sensors fails before a general gamma ray survey is planned. In such a scenario, the self-healing behavior would be the simple action of deleting the sensor from the list of functioning sensors. At the severe end of the range, an example scenario would arise

when the team loses so many workers it can no longer conduct science operations. In this case, the self-healing behavior might be to advise mission control and wait for instructions. In some possible ANTS mission concepts, instead of "calling home" for help, an ANTS team may only need to request a replacement from another team.

ANTS individuals may also have self-healing behaviors. For example, an individual may have the capability of detecting corrupted code (software). In such a case, self-healing behavior would result in the individual requesting a copy of the affected software from another individual in the team, which would enable it to restore itself to a known operational state.

Self-protection. The self protecting behavior of the team will be interrelated with the self-protecting behavior of the individual members. The anticipated sources of threats to ANTS individuals (and consequently to the team itself) will be collisions and solar storms.

Collision avoidance through maneuvering will be limited because ANTS individuals will have limited ability to adjust their orbits and trajectories, since thrust for maneuvering is obtained from solar sails. Individuals will have to coordinate their orbits and trajectories with other individuals to avoid collisions. The main self-protection mechanism for collision avoidance is achieved through planning. The ruler's plans involve constraints that will result in acceptable risks of collisions between individuals.

Another possible ANTS self-protection mechanism could protect against effects of solar storms. Charged particles from solar storms could subject individuals to degradation of sensors and electronic components. When the ruler recognizes that a solar storm threat exists, the ruler would invoke its goal to protect the mission. In addition to its own protection, part of the ruler's response would be to give workers the goal to protect themselves.

As noted in the section on self-configuring behavior, after-effects of protective action will, in general, necessitate ANTS self-reconfiguration. For example, after solar sails had been trimmed for the storm blast of solar wind, individuals will have unplanned trajectories, which will necessitate trajectory adjustments and re-planning and perhaps new goals. Further, in case of the loss of individuals due to damage by charged particles, the ANTS self-healing behavior and the self-optimizing behavior may also be triggered. Thus, there is an interrelatedness of the self-protecting behaviors of the ANTS team and the ANTS individuals.

12.7 Conclusions

NASA missions represent some of the most extreme examples of the need for survivable systems that cannot rely on support and direction from humans while accomplishing complex objectives under dynamic and difficult conditions. Future missions will embody greater needs for longevity in the face of significant constraints, in terms of cost and the safety of human life. Future missions also will have increasing needs for autonomous behavior not only to reduce operations costs and overcome limitations of communications (signal propagation delays and low data rates), but also to overcome the limitations of humans to perform long-term space missions and enable exploration systems to accomplish discovery objectives that otherwise would be infeasible or too costly.

There is an increasing realization that future missions must not only be autonomous but also must exhibit the properties of autonomic systems for the survivability of both individuals and systems. As described, the ACT architecture provides for a flexible implementation of a wide range of intelligent and autonomic agents. The ACT architecture allows for easy removal of components unneeded for reactive agents, and the inclusion of the necessary components to implement intelligent and autonomic agents. It is also flexible

so that additional unforeseen needs can be satisfied by new components that can be added without affecting previous components.

The ultimate goal of our work is to support NASA exploration through technologies that enable survivable systems and evolvable architectures. We see promise in proven agent autonomy and autonomicity technologies, and we see value in transitioning these into operational NASA systems. The implementation of the scenarios discussed above (and others under development) will provide an opportunity to exercise, evaluate, and refine the capabilities supported by the agent architectures. It will also provide an opportunity for space mission designers and developers to "see" agent and autonomic technology in action and the resulting benefits. This will enable them to make a better determination of the role that this technology can play in their missions.

We have illustrated the necessity of autonomicity in space exploration missions with reference to an existing NASA system, namely ACT, and have reiterated its importance not only in typical NASA missions but also for all future missions, including ANTS. It is clear that the dissociation of autonomy and autonomicity as mission characteristics will decrease in the future [10], and that holding that an autonomous system in general does not have the properties of an autonomic system is untenable. Considering autonomic properties to be disjoint is seen to be unsupportable or naïve [8]: examples with relevance to ANTS illustrate that these properties are, in general, interrelated and overlapping.

Acknowledgements

This chapter is based substantially on work reported in [6] and [10].

This work was supported in part by the NASA Office of Safety and Mission Assurance (OSMA) Software Assurance Research Program (SARP) and managed by the NASA Independent Verification and Validation (IV&V) Facility, and by NASA Headquarters Code R. At the University of Ulster, support was provided by the Computer Science Research Institute (CSRI) and the Centre for Software Process Technologies (CSPT) which is funded by Invest NI through the Centres of Excellence Programme, under the EU Peace II initiative.

Bibliography

[1] P. E. Clark, S. A. Curtis, and M. L. Rilee. ANTS: Applying a new paradigm to Lunar and planetary exploration. In *Proc. Solar System Remote Sensing Symposium*, Pittsburgh, Pennsylvania, USA, 20–21 September 2002.

[2] S. A. Curtis, W. F. Truszkowski, M. L. Rilee, and P. E. Clark. ANTS for the human exploration and development of space. In *Proc. IEEE Aerospace Conference*, Big Sky, Montana, USA, 9–16 March 2003.

[3] W. F. Truszkowski, M. G. Hinchey, J. L. Rash, and C. A. Rouff. NASA's swarm missions: the challenge of building autonomous software *IEEE IT Professional*, vol. 6, no. 5, pages 47-52, 2006.

[4] R. Murch. *Autonomic Computing.* IBM Press, 2004.

[5] C. A. Rouff, M. G. Hinchey, J. L. Rash, W. F. Truszkowski, and R. Sterritt. Autonomicity of nasa missions. In *Proc. IEEE International Conference on Autonomic Computing (ICAC'05)*, pages 387–388, Seattle, Washington, USA, 13–16 June 2005. IEEE Computer Society Press, Los Alamitos, Calif.

[6] C. A. Rouff, M. G. Hinchey, J. L. Rash, W. F. Truszkowski, and R. Sterritt. Towards autonomic management of NASA missions. In *Proc. 1st IEEE Workshop on Reliability and Autonomic Management In Parallel and Distributed Systems held in conjunction with 11th International Conference on Parallel and Distributed Systems (ICPADS-2005)*, Fukuoka, Japan, 20–22 July 2005.

[7] R. Sterritt and M. G. Hinchey. Why computer based systems *Should* be autonomic. In *Proc. 12th IEEE International Conference on Engineering of Computer Based Systems (ECBS 2005)*, pages 406–414, Greenbelt, MD, April 2005.

[8] R. Sterritt, C. A. Rouff, J. L. Rash, W. F. Truszkowski, and M. G. Hinchey. Self-* properties in NASA missions. In *4th International Workshop on System/Software Architectures (IWSSA'05) in Proc. 2005 International Conference on Software Engineering Research and Practice (SERP'05)*, pages 66–72, Las Vegas, Nevada, USA, June 27 2005. CSREA Press.

[9] W. Truszkowski and C. Rouff. An overview of the NASA LOGOS and ACT agent communities. In *Proc. World Multiconference on Systemics, Cybernetics and Informatics*, Orlando, Florida, July 2001.

[10] W. F. Truszkowski, M. G. Hinchey, J. L. Rash, and C. A. Rouff. Autonomous and autonomic systems: A paradigm for future space exploration missions. *IEEE Transactions on Systems, Man and Cybernetics, Part C (to appear)*, 2006.

[11] W. F. Truszkowski, J. L. Rash, C. A. Rouff, and M. G. Hinchey. Some autonomic properties of two legacy multi-agent systems — LOGOS and ACT. In *Proc. 11th IEEE International Conference on Engineering Computer-Based Systems (ECBS), Workshop on Engineering Autonomic Systems (EASe)*, pages 490–498, Brno, Czech Republic, May 2004. IEEE Computer Society Press, Los Alamitos, Calif.