NASA Contributions to IT

Swarm Technology at NASA:

Building Resilient Systems

Emil Vassev, Lero—the Irish Software Engineering Research Centre

Roy Sterritt, University of Ulster

Christopher Rouff, Lockheed Martin

Mike Hinchey, Lero—the Irish Software Engineering Research Centre

Developing aerospace systems is a complex task driven by standards and safety requirements to ensure reliability of sophisticated hardware and software. NASA future missions include an approach to developing swarm-based spacecraft systems comprising multiple self-organizing and autonomous spacecraft.

Deep space presents numerous hazards and harsh conditions for remote exploration missions, which must often operate autonomously without intervention from Earth. To increase the survivability of the remote missions, NASA is exploiting principles and techniques that help such systems become more resilient through self-management and automatic adaptation. By adhering to the principles of autonomic computing,[6–8] contemporary spacecraft systems implement vital features for unmanned missions, such as self-configuration, self-healing, self-optimization, and self-protection.

Moreover, biologically inspired approaches target new classes of space exploration missions that use swarm intelligence and swarm cooperation to achieve extremely robust systems. Swarm-based systems comprise thousands of small spacecraft working together to explore places in deep space where a single and monolith spacecraft is impractical.

However, developing such systems—from conceptualization to validation—is a complex multidisciplinary activity, and reliability and safety are key objectives. The systems can't exhibit post-release faults or failures that could jeopardize the mission or cause loss of life. They integrate complex hardware and sophisticated software and thus require careful design and thorough testing to ensure adequate reliability. Moreover, aerospace systems have strict dependability and real-time requirements; need flexible resource reallocation; and must be limited in size, weight, and power consumption.

System engineers thus must optimize their designs for three key factors: performance, reliability, and cost. As a result, the development process, characterized by numerous iterative design and analysis activities, is lengthy and costly. Moreover, for systems requiring certification prior to operation, the control software must go through rigorous verification and validation.

Verification-Driven Software Development

When developing software, it's important to choose a life-cycle process appropriate for the project at hand, because all other activities derive from that process. Aerospace

systems must meet a variety of standards and adhere to high safety requirements, so the software development process for such systems should emphasize verification, validation, certification, and testing (see Figure 1).[1]
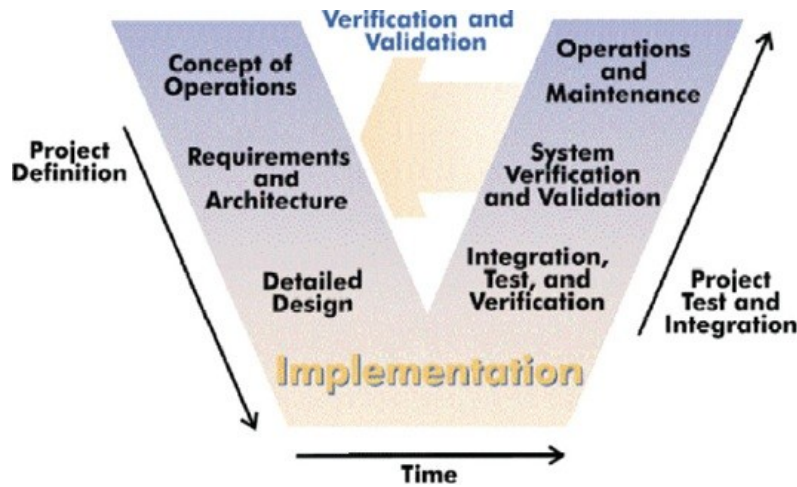

Figure 1. A common NASA software development process.[1] The software development process for such systems should emphasize verification, validation, certification, and testing.

The process should also be technically adequate and cost effective for managing the safety requirements and design complexity and for certifying embedded system software. Most modern aerospace software development projects use some kind of spiral-based methodology over a waterfall process, because it better emphasizes verification.

Emphasizing Safety

The development process should help software developers specify the required level of safety to ensure they design and implement secure systems. NASA engineers can express software safety as a set of features and procedures that ensure predictable system performance under normal and abnormal conditions. Furthermore, when developers specify software safety properly, "the likelihood of an unplanned event occurring is minimized and its consequences controlled and contained."[2]

NASA uses two software safety standards[3] that define four qualitative hazard severity levels—catastrophic, critical, marginal, and negligible—and probability levels—probable, occasional, remote, and improbable. Hazard severity and probability are correlated to derive a risk index for prioritizing risk resolution (see Table 1).

Table 1. NASA's Risk Index Determination.[3]

Hazard severity	Hazard probability			
	Probable	Occasional	Remote	Improbable
Catastrophic	1	1	2	3
Critical	1	2	4	4
Marginal	2	3	4	5
Negligible	3	4	5	5

Dealing with Complexity

Contemporary aerospace systems are designed and implemented as multicomponent systems, where the components are self-contained and reusable, thus requiring high independency and complex synchronization. Moreover, the components of more sophisticated systems are considered as agents (multiagent systems) incorporating some degree of intelligence. Intelligent agents are key to realizing self-managing systems.[4]

Developing aerospace systems usually involves

- multicomponent systems, where you can't always model intercomponent interactions and system-level impact;
- elements of AI;
- autonomous systems;
- evolving systems;
- high-risk and high-cost systems, often intended to perform missions with significant societal and scientific impacts;
- rigid design constraints;
- the potential for extremely tight design space;
- highly risk-driven systems, where you can't always capture or understand the risks and uncertainty.

For the development of multicomponent systems with elements of AI and autonomous behavior, NASA is investigating concepts from swarm computing.

Swarm Computing

Swarm computing is a biology-inspired approach to decentralized systems composed of relatively simple agents that can self-organize to solve complex problems together through direct or indirect interactions.[9] It emphasizes a special form of swarm intelligence, a behavioral metaphor for solving distributed problems on the basis of the principles underlying the behavior of natural multiagent systems, such as ant colonies and bird flocks.

Swarm intelligence provides a new behavioral model for multiagent systems stemming from local interactions between individuals with simple rule sets and no global knowledge. Coherence and cooperation emerge from a global viewpoint without any active push for it at the individual level. Swarm computing applications include optimization algorithms, communications networks, and robotics. Applying swarm intelligence to robotic devices lets the individual members of the swarm exhibit independent intelligence.

Scientists began investigating swarm intelligence in the 1950s, when they started to study how social animals and insects communicate and coordinate themselves. French biologist Pierre-Paul Grassé, a pioneer in this field, studied structured approaches in the otherwise seemingly chaotic nest-building process of termites.[10] This area of research gained popularity with research projects on multiagent systems in 1970s. Nowadays, NASA is investigating the application of swarm computing to both spacecraft and surface-based rovers. The Autonomous Nano Technology Swarm (ANTS) concept mission is a collaboration between NASA Goddard Space Flight Center and NASA Langley Research Center. It aims to develop revolutionary mission architectures and exploit AI techniques and paradigms in future space exploration.

Autonomous Nano Technology Swarm

The ANTS mission has several different concepts and goals.[11] For example, the Super Miniaturized Addressable Reconfigurable Technology concept exploits the use of tetrahedrons to build small reconfigurable robots that are combined to form swarms, while the Saturn Autonomous Ring Array would use thousands of pico-class spacecraft, organized as 10 subswarms (each with specialized instruments) to perform in-situ exploration of Saturn's rings.

Additionally, lunar-base applications aim to exploit new NASA-developed miniaturized robotic technologies as the basis of moon landers launched from remote sites. The activities would exploit innovative techniques to let rovers move in an amoeboid-like fashion over the moon's uneven terrain.

Finally, the Prospecting Asteroid Mission (PAM) would launch thousands of pico-class spacecraft to explore the asteroid belt and collect data on asteroids of interest. This novel approach to asteroid belt resource exploration (see Figure 2) requires significant autonomy, minimal communication with Earth, and a set of very small explorers with few consumables.[15] These explorers are pico-class, low-power, and low-weight spacecraft units, yet they can operate as fully autonomous and adaptable agents. Each spacecraft is equipped with a solar sail and relies primarily on power from the sun, using only tiny thrusters to navigate independently. Each spacecraft also has onboard computation, AI, and heuristics systems for control at the individual and team levels. By forming a swarm, the spacecraft can interact with each other and self-organize.
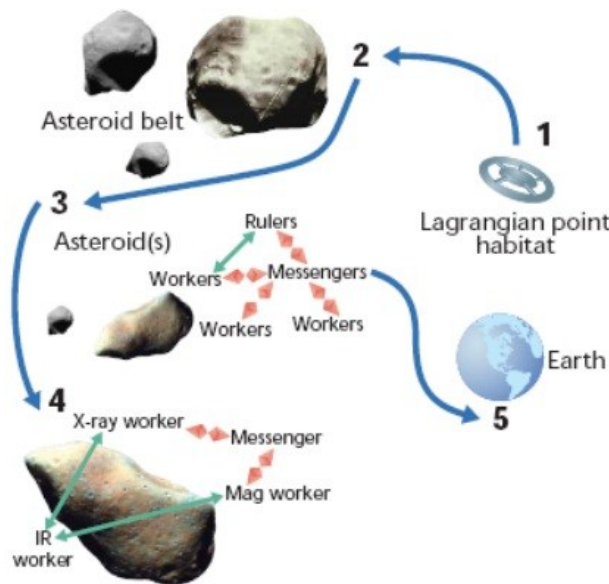

Figure 2. The Autonomous Nano Technology Swarm (ANTS) project will exploit AI techniques and paradigms in future space exploration.[15] In particular, the Prospecting Asteroid Mission aims to launch 1,000 pico-class spacecraft to explore the asteroid belt.

In general, a swarm consists of several subswarms, which are temporal groups organized to perform a particular task. Each swarm group has a group leader (*ruler*), one or more *messengers*, and a number of *workers* carrying a specialized instrument (see

Figure 2). The messengers connect the team members when they can't connect directly because of long distance or a barrier.

Autonomic Properties

For ANTS exploration, individual autonomy isn't crucial, but the mission can't succeed unless each team has the following autonomic properties.[16]

Self-configuration. ANTS must adapt to changes in the system. Moreover, ANTS must be fully reconfigurable to support concurrent exploration and examination of hundreds of asteroids. Reconfiguration might also be required in the event of a failure or some other anomaly.

Self-optimizing. ANTS must improve their performance on the fly. Leaders can use the gained experience to self-optimize, thus improving their ability to identify asteroids. Messengers, strive to find the best position to improve the communication among the swarm units. Workers also self-optimize through learning and experience.

Self-healing. ANTS must recover from errors or damage, including those caused by damage due either to a solar storm or to a collision with an asteroid or another spacecraft.

Self-protecting. ANTS must anticipate and cure intrusions. For example, ANTS must protect itself from solar storms, where charged particles can degrade sensors and electronic components and destroy solar sails.

Smart Self-Healing in ANTS

Self-healing in ANTS is about finding the right strategy for repairing faulty spacecraft units without decreasing the overall swarm performance or affecting mission goals. A smart self-healing strategy[23] distributes a repair plan among the spacecraft units participating in the self-healing process.

Repairing a spacecraft unit is usually a self-task that the faulty unit performs, but it could also involve other spacecraft units. In such a case, a ruler or an idle worker must drive the self-healing process and assign the self-healing actions to the participating spacecraft units. A smart self-healing strategy might decide that it's not worth repairing a faulty unit—it might be better to destroy or transform it.[25,27]

For example, a faulty worker could use self-healing actions to transform itself into a ruler or messenger, which is possible because the ANTS spacecraft units are built from reusable components. Or, when such transformations aren't possible, the worker could transform itself into a stand-by "shield."[27] A shield unit sails nearby and strives to protect the replacement worker from different hazards. For example, a shield unit could take the impact of an asteroid about to hit the replacement worker.

A smart self-healing unit will make this sacrifice to increase the swarm's overall performance and survivability by minimizing the time and resources needed to recover from a probable impact involving an active worker, ruler, or messenger. When a unit is damaged such that it can no longer move, it eventually will self-destruct (not necessarily physically but possibly through removal from the swarm), thus avoiding the risk of collision with another ANTS unit.[25]

Awareness and Resilience

The concept of awareness plays a crucial role in designing resilient spacecraft systems. Conceptually, awareness is a product of knowledge and monitoring. A spacecraft unit maintains knowledge to track changes in the unit itself and, to some extent, changes in the swarm and the surrounding space environment. Thus, an ANTS mission must also maintain special situational knowledge, eventually expressed as patterns intended to cover special situations or relevant changes in the space environment, swarm system, or individual spacecraft. For example, a situation could be a fault occurrence detected as a fault—that is, unobservable or unexpected event.

A common ANTS awareness model comprises a special awareness control loop that reveals four distinct functions:

- monitoring,
- recognition,
- assessment, and
- learning.

The mission should monitor the individual spacecraft units and space environment using sensors and communication links to collect, aggregate, filter, manage, and report on internal and external information. It should use knowledge to recognize and track changes in individual spacecraft units, the entire swarm (ANTS), or the environment (context). Its assessments determine points of interest, generate hypotheses about situations involving these points, and recognize situational patterns. Finally, it should be able to *learn* by generating new situational patterns and maintaining a history of property changes. These four functions—inspired by the IBM Monitor, Analyze, Plan, and Execute control loop[6] and by self- and environmental-awareness (later referred to as self-situation) conceptual control loops[17]—help make ANTS aware of internal and external changes and situations.

Note that ANTS should rely on both proactive and reactive monitoring.[7] Reactive monitoring happens after an event (external or internal) has occurred; proactive monitoring relies analyzing data collected via sensors and input communication links to raise alerts before the event happens. In addition to the specific control and notification messages, to facilitate proactive monitoring, the individual spacecraft units regularly exchange *pulse-beat messages*,[18] which carry useful information (such as the sender's current health status).[19] For example, each worker regularly sends pulse-beat messages to its group's ruler. This helps the ruler determine when a worker can't continue operating because of a failure.[8]

Formal Methods for ANTS Development

Software reliability is crucial to successful software development. ANSI defines software reliability as "the probability of failure-free software operation for a specified period of time in a specified environment."[20] Practice has shown that traditional development methods can't guarantee software reliability and prevent software failures. Moreover, software developed using formal methods tends to be more reliable.

Formal methods help ensure the quality of aerospace systems, in which system failures can easily cause safety hazards. Modern formal methods provide a computer system development approach that provides both a formal notation and suitable mature

tool support.[5] The formal notation specifies the requirements or models a system design in a mathematical logic, while the tool support helps demonstrate that the implemented system meets its specification.

Even if a full proof is hard to achieve in practice because of engineering and cost limitations, it makes software and hardware systems more reliable. By using formal methods appropriately within the software development process, aerospace system developers can reduce overall development costs.[5] In fact, the costs of formal methods tend to be high early in the system life cycle but lower in the coding, testing, and maintenance stages, where error correction is far more expensive.

Developers have successfully used formal approaches on algorithms for resilient behavior in ANTS based on context awareness and self-awareness.[21,22] Considering the challenge of self-management, the best approach to developing reliable and robust ANTS systems is to use formal methods dedicated to autonomic computing, such as Autonomic System Specification Language[23] or Requirements-to-Design-to-Code.[24] We have successfully used ASSL to specify autonomic features and generate prototype models for ANTS, such as self-configuring, self-healing, self-scheduling, and emergent self-adapting models.[27]

Current verification methods aren't sufficient to verify adaptive systems such as ANTS, because they don't scale to support the huge state space of such systems. This obstacle is partially overcome through the automatic code generation that ASSL enables. This helps developers easily generate multiple prototypes of the system and perform thorough testing of the autonomic features.

Robotic technology such as swarm robotics missions, automatic probes, and unmanned observatories allow for space exploration without risking human lives. Swarm technologies hold promise for exploration and scientific missions that require capabilities unavailable to missions designed around single spacecraft. Although swarm autonomy is clearly essential for missions where human control isn't feasible, individual autonomy is essential for survival of individual spacecraft as well as the entire swarm in hostile space environments.

The derivative benefits of swarm computing require advances in miniaturization and nanotechnology. Moreover, the need for more efficient on-board power generation and storage motivates research in solar energy and battery technology, and the need for energy-efficient propulsion motivates research on solar sails and other technologies, such as electric-field propulsion.

Groundbreaking advances in swarm intelligence research at NASA and other organizations (both civilian and military) show great possibilities for applying swarm computing beyond just space exploration. In the near future, swarms of nanobots operating in the human body should be able to discover and kill cancer tumors. Swarms of tiny and intelligent submarines will explore the ocean floor and clean up the marine bays. Swarms of military robots will perform military operations, such as a convergent attack on targets from multiple axes.

Acknowledgements

References

1. C. Philippe, "Verification, Validation, and Certification Challenges for Control Systems," *The Impact of Control Technology*, T. Samad and A.M. Annaswamy, eds., IEEE Control Systems Soc., 2011, 205-206.

2. D.S. Herrmann, *Software Safety and Reliability*, IEEE CS Press, 1999.

3. *Software Safety*, NASA Technical Standard NASA-STD-8719.13A, NASA, 1997.

4. D. Gilbert et al., "IBM Intelligent Agent Strategy," white paper, IBM Corp., 1995.

5. M. Hinchey, J. Bowen, and E. Vassev, "Formal Methods," *Encyclopedia of Software Eng.*, P.A. Laplante, ed., Taylor & Francis, 2010, pp. 308–320.

6. J.O. Kephart and D.M. Chess, "The Vision of Autonomic Computing," *Computer*, vol. 36, no. 1, 2003, pp. 41–50.

7. R. Sterritt, "Autonomic Computing," *Innovations in Systems and Software Eng*., vol. 1, no. 1, 2005, pp. 79–88.

8. R. Sterritt and M. Hinchey, "Engineering Ultimate Self-Protection in Autonomic Agents for Space Exploration Missions," *Proc. 12th IEEE Int'l Conf. and Workshops on the Engineering of Computer-Based Systems* (ECBS 05)*, 2005, pp. 506–511.

9. E. Bonabeau and G. Theraulaz, "Swarm Smarts," *Scientific Am*., Mar. 2000, pp. 72–79.

10. P. Grasse, *Termitologia*, Tome II, Fondation des Sociétés, 1984.

11. W. Truszkowski et al., Autonomous and Autonomic Systems: With Applications to NASA Intelligent Spacecraft Operations and Exploration Systems, NASA Monographs in Systems and Software Eng. series, Springer, 2010.

12. M.T. Morrow, C.A. Woolsey and G.M. Hagerman, Jr, "Exploring Titan with Autonomous, Buoyancy-Driven Gliders," *J. British Interplanetary Soc.*, vol. 59, no. 1, 2006, pp. 27–34.

13. T. Iida, J.N. Pelton and E. Ashford, *Satellite Communications in the 21st Century: Trends and Technologies*, Am. Inst. of Aeronautics and Astronautics, 2003.

14. M.G. Hinchey, R. Sterritt, and C. Rouff, "Swarms and Swarm Intelligence," *Computer*, vol. 40, no. 4, 2007, pp. 111–113.

15. W. Truszkowski et al., "NASA's Swarm Missions: The Challenge of Building Autonomous Software, *IT Professional*, vol. 6, no. 5, 2004, pp. 47–52.

16. R. Sterritt et al., "Next Generation System and Software Architectures: Challenges from Future NASA Exploration Missions," *Science of Computer Programming*, vol. 61, no. 1, 2006, pp. 48–57.

17. R. Sterritt and M.G. Hinchey, "From Here to Autonomicity: Self-Managing Agents and the Biological Metaphors that Inspire Them," *Proc. Integrated Design & Process Technology Symp*. (IDPT 05), 2005, pp. 143–150.

18. R. Sterritt, "Towards Autonomic Computing: Effective Event Management," *Proc. 27th Ann. IEEE/NASA Software Eng. Workshop* (SEW), IEEE CS Press, 2002, pp. 40–47.

19. R. Sterritt and D.W. Bustard, "Towards an Autonomic Computing Environment," *Proc. IEEE DEXA 2003 Workshops—First Int'l Workshop on Autonomic Computing Systems*, 2003, pp. 694–698.

20. ANSI/IEEE: Standard Glossary of Software Engineering Terminology, STD-729-1991, ANSI/IEEE, 1991.

21. E. Vassev, M. Hinchey and P. Nixon, "A Formal Approach to Self-Configurable Swarm-Based Space-Exploration Systems," *Proc. 2010 NASA/ESA Conf. Adaptive Hardware and Systems* (AHS 10), IEEE CS Press, 2010, pp. 89–96.

22. E. Vassev and M. Hinchey, "Self-Awareness in Autonomous Nano-Technology Swarm Missions" *Proc. 5th IEEE Conf. Self-Adaptive and Self-Organizing Systems Workshops* (SASOW 11), IEEE CS Press, 2011, pp. 133–

23. E. Vassev, "Towards a Framework for Specification and Code Generation of Autonomic Systems," doctoral dissertation, Dept. of Computer Science and Software Engineering, Concordia Univ., 2008.

24. M. Hinchey, J. Rash, and C. Rouff, *Requirements to Design to Code: Towards a Fully Formal Approach to Automatic Code Generation*, tech. report TM-2005-212774, NASA Goddard Space Flight Center, 2004.

25. R. Sterritt and M.G. Hinchey, "Apoptosis and Self-Destruct: A Contribution to Autonomic Agents?" *Proc. Third NASA-Goddard/IEEE Workshop on Formal Approaches to Agent-Based Systems* (FAABS III), LNAI 3228, Springer-Verlag, 2004, pp. 262–270.

27. J. Pena, M.G. Hinchey, and R. Sterritt, "Towards Modeling, Specifying and Deploying Policies in Autonomous and Autonomic Systems Using an AOSE Methodology," *Proc. 3rd IEEE Int'l Workshop Eng. of Autonomic and Autonomous Systems* (EASe 06), IEEE CS Press**,** 2006, pp. 37–46.

28. E. Vassev and M. Hinchey, "An Emergent Self-Adapting Behavior Model for NASA Swarm-Based Exploration Missions," *Proc. Second IEEE Int'l Conf. Self-Adaptive and Self-Organizing Systems* (SASO 08), IEEE CS Press, 2008, pp. 473–474.

Emil Vassev is a research fellow at Lero—The Irish Software Engineering Research Centre at the University of Limerick, Ireland. His research interests include knowledge representation and self-awareness for self-adaptive systems and software development methodologies for autonomic systems. Vassev received his PhD in computer science from Concordia University, Montreal, Canada. He's a member of IEEE Computer Society and the Irish Computer Society. Contact him at emil.vassev@lero.ie.

Roy Sterritt is a faculty member at the University of Ulster and a researcher within the Computer Science Research Institute (CSRI). His research interests include autonomic computing, autonomic communications, and apoptotic computing. Sterritt received his masters in business strategy from University of Ulster. He is a member of IEEE and IEEE Computer Society. He's a Founding Chair of the IEEE Technical Committee on Autonomous and Autonomic Systems. Contact him at r.sterritt@ulster.ac.uk.

Chris Rouff is a manager at Lockheed Martin Advanced Technology Laboratories. His research interests include swarm-based and emergent systems, formal verification, and verification of adaptive systems. Rouff received his PhD in computer science from the University of Southern California. Contact him at christopher.rouff@lmco.com.

Mike Hinchey is director of Lero—The Irish Software Engineering Research Centre and professor of software engineering at the University of Limerick, Ireland. He's also a NASA expert consultant. His research interests include formal methods, self-managing software, and evolving critical systems/. Hinchey received his PhD in computer science from the University of Cambridge. He's a senior member of IEEE. Contact him at mike.hinchey@lero.ie.

Swarm-Computing Projects at NASA

Aside from the Autonomous Nano Technology Swarm project (discussed in the main text), NASA has several other swarm-computing projects, run at Virginia Institute of Technology and funded by the NASA Institute for Advanced Concepts. The intent is to develop low-cost planetary exploration systems that could run autonomously for years in

harsh environments, such as in the sulfuric acid atmosphere of Venus or on Titan (the largest of Saturn's moons).[12] A NASA swarm-related project, "Extremely Large Swarm Array of Picosats for Microwave/RF Earth Sensing, Radiometry, and Mapping," proposes a specialized telescope designed to characterize soil moisture content, atmospheric water content, and snow accumulation levels.[13] The telescope could also monitor flooding, help provide emergency management after hurricanes, help with weather and climate prediction, and identify geological features.[13]

NASA is also studying swarm intelligence for communication network routing. For example, swarm-based routing algorithms can solve global optimization and resource allocation problems in NASA's Earth orbit satellite constellation networks and the Mars networks.[14]

References

12. M. T. Morrow, C.A. Woolsey and G.M. Hagerman, Jr., "Exploring Titan with Autonomous, Buoyancy-Driven Gliders," *J. British Interplanetary Society*, vol. 59, no. 1, 2006, pp.27–34.

13. T. Iida, J.N. Pelton and E. Ashford, Satellite Communications in the 21st Century: Trends and Technologies, AIAA, 2003.

14. M.G. Hinchey, R. Sterritt, and C. Rouff, "Swarms and Swarm Intelligence," *Computer*, vol.40, no.4, 2007, pp. 111–113.