

Toward Dependable Swarms

Alan FT Winfield

Intelligent Autonomous Systems Laboratory,
UWE Bristol, Coldharbour Lane, Bristol BS16 1QY, UK
Alan.Winfield@uwe.ac.uk,
WWW home page: <http://www.ias.uwe.ac.uk/>

Abstract. This review paper sets out to explore the question of how future complex engineered systems based upon the swarm intelligence paradigm could be assured for dependability. The paper introduces the new concept of 'swarm engineering': a fusion of dependable systems engineering and swarm intelligence. The paper reviews the disciplines and processes conventionally employed to assure the dependability of conventional complex (and safety critical) systems in the light of swarm intelligence research and in so doing tries to map processes of analysis, design and test for safety-critical systems against relevant research in swarm intelligence. The paper concludes that while there is much work to be done before dependable swarms become a reality there are a number of promising directions in mathematical modelling for proving that a system exhibits desirable behaviours.

1 Vision

From an engineering standpoint the design of complex distributed systems based upon swarm intelligence is compellingly attractive but problematical. A distinguishing characteristic of distributed systems based upon swarm intelligence is that they have no hierarchical command and control structure, and hence no common mode failure point or vulnerability. Typically, individual agents make decisions autonomously, based upon local sensing and communications [3][4]. Systems with these characteristics could, potentially, exhibit very high levels of robustness, in the sense of tolerance to failure of individual agents; much higher levels of robustness than in complex distributed systems based on traditional design approaches. However, that robustness comes at a price. Complex systems with swarm intelligence might be very difficult to control or mediate if they started to exhibit unexpected behaviours. Such systems would therefore need to be designed and validated for a high level of assurance that they exhibit intended behaviours and *equally importantly* do not exhibit unintended behaviours. It seems reasonable to assert that future engineered systems based on the swarm intelligence paradigm would need to be subject to processes of design, analysis and test no less demanding than those we expect for current complex systems.

Some might argue that a 'dependable swarm' is an oxymoron; that the swarm intelligence paradigm is intrinsically unsuitable for application in engineered systems that require a high level of integrity. The idea that overall desired swarm behaviours are not explicitly coded anywhere in the system, but are instead an emergent consequence of the interaction of individual agents with each other and their environment, might appear to be especially problematical from a dependability perspective. This paper suggests that this is not so: that systems which employ emergence should, in principle, be no more difficult to validate than conventional complex systems and, indeed, that some characteristics of swarm intelligence are highly desirable from a dependability perspective.

The aim of this paper is to explore the question of how future engineered systems based on the swarm intelligence paradigm might be designed, analysed and tested for dependability. The paper attempts to do this by the juxtaposition of two hitherto disconnected disciplines: dependable systems engineering and the design of multi-agent systems based on the swarm intelligence paradigm (which, for brevity, we shall term 'swarm engineering'). This is a big question, a complete answer to which is well beyond the scope of this paper. The paper instead tries to set out the important questions for the ongoing study of dependable swarms.

This paper proceeds as follows. Section 2 is a review of current best practice in the field of dependable systems engineering. While outlining and referencing the processes and methodologies of analysis, design and test, the paper will reflect on what these might mean, in practice, for swarm engineering. Section 3 then concludes with a discussion and outlook.

2 Dependable Swarm Engineering

Current best practice in assuring the dependability of complex systems requires that a set of processes and disciplines are transparently applied during system *analysis*, *design* and *test*, see Anderson et al [1]. This paper now considers the approaches that would typically need to be applied to safety-critical systems in the context of swarm engineering, under these three headings. Note that best practice requires that the processes of analysis, design and test are applied concurrently and iteratively, so the ordering of the following sections should not be taken to imply sequence.

2.1 Analysis

From a dependability perspective, analysis is concerned with trying to establish two properties of a system: 'liveness' and 'safety'. Liveness is defined as the property of exhibiting desirable behaviours (doing the right thing) and safety is defined as the property of not exhibiting undesirable behaviours (not doing the wrong thing). While these properties are clearly somewhat complementary proof of one does not imply proof of the other, by inversion. A system that is provably safe could, for example, do the wrong thing safely. Although it may appear counter-intuitive, the methods needed to verify these two properties are not the same.

Verification of liveness Verification of 'liveness' requires that we formally prove that a system exhibits desirable behaviours. Conventionally this requires analytical or mathematical modelling. In the safety systems community the use of testing alone to prove liveness is now deprecated on the grounds that systems are becoming too complex to allow anything like acceptably complete test coverage, or even to allow complete test specifications to be written. Simulation is similarly regarded as unacceptable as an analysis tool (an interesting observation given the widespread use of simulation within swarm intelligence research¹). Simulation is nevertheless accepted as a useful tool in prototyping, to for instance refine the system specification and to understand the design or parameter space.

Complete verification of liveness thus requires mathematical modelling at two levels: the individual agent, and the swarm as a whole.

Let us first consider the individual agent. Often, single artificial agents within swarms are designed using the behaviour-based control paradigm. Behaviour-based control is appropriate given that such agents are often reactive finite-state machines with relatively few states (behaviours). Harper and Winfield have developed a methodology, based on a second order extension of Lyapunov stability theorems, proving both marginal and asymptotic stability [8]. The significance of second order stability is that position control in mobile agents can generally only be achieved through actuators that generate forces which govern acceleration; the second derivative of position. Of particular significance is that these second order stability theorems provide an explicit mathematical representation of subsumption [5]. Based on this observation Harper has developed a rigorous design methodology called 'Direct Lyapunov Design' which leads from analysis directly to a colony-style behaviour based controller [6] which is provably stable (in the sense of Lyapunov), and exhibits the liveness property, see Harper, 2004 [9]. This methodology thus advantageously encompasses both analysis and design. We also conjecture that this approach could be extended to cover the analysis of hazard states as well as goal states, thus offering the possibility of verifying liveness and safety with a single analysis, see Appendix A.

Now consider the mathematical modelling of the whole swarm. There has been relatively little work in this direction, but one very promising approach is the probabilistic model developed by Martinoli et al, 1999 [15]. In this approach the interactions of agents with each other and their environment are modelled as a series of stochastic events, with probabilities determined by simple geometrical analysis. By modelling several series together, one for each agent, the overall behaviour of the swarm can be studied. The approach of Martinoli et al may be thought of as bottom up (or microscopic as they describe it). A top down (or macroscopic) approach has been developed by Lerman and Galystan, 2001 [11]. Like Martinoli, Lerman and Galystan regard the behaviour of each agent

¹ For a valuable discussion of the role of simulation in embodied systems research see Ziemke, 2003 [22].

as inherently probabilistic and Markovian, because their next state is a function only of their current state. However, they develop an overall model of the system using the stochastic Master Equation (from stochastic dynamical systems), then derive rate (differential) equations from it, which describe how the average macroscopic system properties change over time.

Verification of Safety To verify 'safety' we need to prove that a system does not exhibit undesirable behaviours. In order to attempt such a proof first requires that we identify and articulate all possible undesirable behaviours. This is called 'hazard analysis' and is problematical with conventional complex systems; and there is no reason to suppose that identifying the hazards in swarm engineered systems will be any different. Hazards analysis is problematical because there are no formal methods for identifying hazards. It simply has to be done by inspection (typically by 'extreme brainstorming' to try and list all possible hazards no matter how seemingly implausible or improbable).

Given a reasonably well understood operational environment there are two reasons for undesirable behaviours: random errors, or systematic (design) errors. Random errors are those due to hardware or component faults, and these are typically analysed using techniques such as Failure Modes and Effect Analysis (FMEA). The likelihood that random errors cause undesirable behaviours can be reduced, in the first instance, by employing high reliability components. But systems that require high dependability will typically also need to be fault tolerant, through redundancy for example. This is an important point since swarm engineered systems should, in this respect, offer significant advantages over conventional complex systems. Two characteristics of swarms work in our favour here. Firstly, simple agents with relatively few rules lend themselves to FMEA, and their simplicity facilitates design for reliability. Secondly, swarms consist of multiple agents and hence, by definition, exhibit high levels of redundancy and tolerance to failure of individual agents. Indeed, swarms may go far beyond conventional notions of fault tolerance by exhibiting tolerance to individuals who actively thwart the overall desired swarm behaviour.

Systematic errors are those aspects of the design that could allow the system to exhibit undesirable behaviours. For swarm engineered systems analysis of systematic errors clearly needs to take place at two levels: in the individual agent and for the swarm as a whole. Analysis of systematic errors in the individual agent should be helped by the relative simplicity of the agents, but is not trivial. In general terms we would need to prove that an agent's state-space trajectory is always 'away from' the hazard states, as mentioned in section 2.1.1 and the conjecture of Appendix A. Analysis of systematic errors for the swarm as a whole is much more problematical, particularly if the desired behaviours are emergent. Proof of safety for the overall swarm would appear to require that we prove that there are no undesired emergent behaviours. How to prove this to an acceptable level of confidence is by no means clear.

2.2 Design

The design of systems based on the swarm intelligence paradigm is challenging, not least because there are no principled design approaches for determining the behaviours required of the individual agents in order to give the desired emergent overall swarm behaviour. Indeed, some would argue that a principled approach to the design of emergence is impossible. This paper is however concerned with dependability, and there is no reason to suppose that emergent behaviours cannot form part of a dependable system.

Most complex systems are designed top-down from an overall functional design specification (FDS), by functional decomposition: breaking down the overall system into smaller and smaller components, then defining each of those components and the interfaces between them. What differentiates design for dependable, or safety critical, systems is that it will typically use a structured design methodology to provide a framework for capturing and documenting the design as it progresses, top down. The Yourdon structured design methodology, for instance, is based upon the dataflow paradigm. It starts at the top level by describing the overall system and its interfaces with its operational environment as a 'context diagram': this is level 0. The context diagram is then decomposed into level 1 'processes' and the dataflows between them, expressed in a data flow diagram (DFD). Each process in level 1 is then further decomposed into lower level DFDs, and so on, see Yourdon, 1989 [21]. The structured design may well be applied within the discipline of a document driven approach [10], together with code inspection [7].

If we consider the applicability, and utility, of the Yourdon structured design methodology to swarm engineered systems it is clear that, at the top level, we can express the single swarm and its interfaces with the environment as a context diagram (level 0). Equally well, we could describe the internal processes of an individual agent with a data flow diagram (level 2). What is interesting, however, is how we might express the intermediate level 1 as a DFD. If we assume that single agents are (a) mobile, and (b) able to sense only their immediate neighbours [14][19], then the level 1 DFD will reflect the instantaneous topography of the swarm. After the mobile agents have moved, the DFD must change to reflect the new swarm topography. This interestingly suggests an extension of the DFD which we could term the 'dynamic data flow diagram'.

2.3 Test

Within the safety critical systems community there is general agreement that testing, whilst essential, can only provide a limited measure of confidence in the liveness and safety properties of a system. There are two problems. Firstly, to write a complete test specification for a complex system is very difficult, and secondly to achieve 100% test coverage (which means exercising every possible execution path through control code, or state machines, under controlled conditions), whilst not technically impossible, is infeasibly time consuming for even moderately complex systems. Thus even the most safety critical systems in use

today, such as aircraft flight management systems, will have been put through demanding but ultimately incomplete testing [12]. This is the reason that testing needs to go hand in hand with mathematical modelling, as discussed in 2.1 above.

Typically, a test regime for safety critical systems is split into two parts: system level functional testing and component level testing. System level testing is primarily concerned with liveness, and typically treats the overall system as a 'black box', testing only for correct behaviour of the system as a complete entity against a system test specification (STS). Component level testing breaks the system into its sub-systems and tests each one individually. Thus component level testing is the equivalent of system level 'white box' testing.

At component level, sub-systems need to be tested functionally. This normally requires that 'test harnesses' are created to enable components to be tested in isolation from the rest of the system. A test harness will set up input conditions for a component that might be extremely difficult to create by treating the system as an integrated whole. Test coverage can be measured directly in a process called 'dynamic analysis', which 'instruments' code such that each time it is executed a tally is kept of the number of times every possible execution path has been exercised. Dynamic analysis is an iterative (and cumulative) process in which ever more ingenious new tests are devised (typically by inspection of the code), in order to exercise those parts of the code revealed to have been not executed by the testing so far. The process continues until the target level of test coverage has been achieved. Needless to say dynamic analysis is a difficult and time consuming process. For completeness 'static analysis' should also be mentioned since it often goes hand in hand with dynamic analysis. Static analysis measures code without actually executing it against coding standards including, typically, the McCabe complexity measure to assess the 'spaghetti-ness' of code [16].

If we now consider swarm engineered systems in the light of the discussion above, it is clear that system level testing needs to apply to the swarm as a whole, operating in its intended environment, and component level testing applies, in effect, to an individual agent. The fact that individual agents are often identical in swarm systems, and relatively simple in functional terms, suggests that component level testing should not be intractable. This view is, however, probably illusory, since the 'environment' for a single agent is the sum total of the other (presumably) neighbouring agents and the environment. Complete testing of a single agent would require that every possible configuration of neighbours and environment is specified, and repeatable tests devised (the neighbours plus environment becomes in effect the test harness). Again there has been little work in mobile robotics to quantitatively assess the effect of its environment on an individual robot, see Schner et al, 1995 [17]; Smithers, 1995 [18]. The recent paper of Nehmzow and Walker (2003) suggests methods based on dynamical systems theory, time series analysis and deterministic chaos theory [13].

The question of how to write a system test specification (STS) for the swarm as a whole might appear to be problematical given that the internal structure

of the swarm is typically highly dynamic and chaotic. However, if we discipline ourselves to treating the swarm as a single entity then it should be possible to develop tests for the desired swarm behaviours. These will almost certainly be statistical, measuring for instance the frequency with which a given behaviour reaches a quantitative threshold condition of achievement within a given time frame, over repeated test runs. Thus, developing an STS for a swarm engineered system is likely to require careful attention to defining criteria for the achievement of swarm behaviours, including metrics for swarm properties such as mean swarm velocity, or mean area coverage.

3 Discussion and Outlook

This paper has attempted a juxtaposition of dependable systems engineering with swarm intelligence and in so doing has tried to map processes of analysis, design and test for safety-critical systems against relevant work in swarm intelligence research. Perhaps not surprisingly, there is not a great deal of overlap between the two fields. To the author's knowledge there has not been, to date, a single real-world application of swarm engineering with real physical agents. Thus no-one has yet had to face the challenge of assuring the dependability of such a system.

The analysis of this paper is revealing, both in exposing the extent of the work to be done to explore the issues of validating swarm engineered systems, but also in highlighting a number of promising analytical approaches that would appear to offer models for proving that a system exhibits the desired behaviours (liveness). Although these approaches are still evolving, it is possible that for the class of stochastic, Markovian systems, employing individual agents with behaviour-based control, we could both prove stability within individual agents and at the same time construct a useful model of overall system properties.

4 Acknowledgements

The author gratefully acknowledges discussions with Chris Harper, Chris Melhuish and Julien Nembrini during preparation of this paper.

5 Appendix A

The work of Harper, 2004 [9], shows that if we have a behaviour-based controller in which behaviours are implemented as motor schema, then we can prove that the Euclidian distance $\|\underline{x} - \underline{x}_g\|$ is a Lyapunov function for that schema and therefore that its behaviour is stable with respect to the goal states \underline{x}_g . This represents a formal proof of 'liveness'.

Conjecture: that there is a Lyapunov function $V(\underline{x})$ defined as the ratio of the Euclidian distance of the goal states \underline{x}_g and the hazard states \underline{x}_h ,

$$V(\underline{x}) = \frac{\|\underline{x} - \underline{x}_g\|}{\|\underline{x} - \underline{x}_h\|} \quad (1)$$

and if the trajectory of $V(\underline{x})$ is negative, i.e. $\dot{V}(\underline{x}) < 0$ then the agent will both seek its goals and avoid its hazards at the same time. In other words the liveness and safety properties are stable over state space.

References

1. Anderson T, Avizienis A and Carter WC: Dependability: Basic Concepts and Terminology, Series: Dependable Computing and Fault-Tolerant Systems Volume 5, Laprie, J-C (ed), Springer-Verlag, New York (1992)
2. Bennett P: Software Development for the Channel Tunnel: A Summary, Journal of High Integrity Systems, **1**(2) (1994) 213-220
3. Bonabeau E, Dorigo M, and Théraulaz G: Swarm Intelligence: from natural to artificial systems, Oxford University Press (1999)
4. Bonabeau E and Théraulaz G: Swarm Smarts, Scientific American, March (2000) 72-79
5. Brooks RA: Cambrian Intelligence: the Early History of the New AI, MIT Press (2000)
6. Connell JH: Minimalist mobile robotics: a colony-style architecture for an artificial creature, Academic Press Professional, San Diego (1990)
7. Fagan ME: Design and Code Inspections to Reduce Errors in Program Development, IBM Systems Journal, **15**(3) (1976)
8. Harper C and Winfield A: Direct Lyapunov Design - A Synthesis Procedure for Motor Schema Using a Second-Order Lyapunov Stability Theorem, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, Lausanne, October (2002)
9. Harper C: A Rational Methodology for Designing Behaviour Based Systems for Safety Related Applications, PhD Thesis, University of the West of England, Bristol (2004)
10. Institution of Electrical Engineers: Guidelines for the documentation of computer software for real time and interactive systems, IEE London, 2nd Edition (1990)
11. Lerman K and Galstyan A: A General Methodology for Mathematical Analysis of Multi-Agent Systems, USC Information Sciences Technical Report ISI-TR-529, (2001)
12. Littlewood B and Thomas M: Reasons why Safety-Critical Avionics Software cannot be Adequately Validated, Proc. 1st UK Safety Systems Symposium, Springer-Verlag (1993)
13. Nehmzow U and Walker K: The Behaviour of a Robot is Chaotic, AISB Journal **1**(4) (2003) 373-388
14. Nembrini J, Winfield A and Melhuish C: Minimalist Coherent Swarming of Wireless Connected Autonomous Mobile Robots, Proc. Simulation of Artificial Behaviour '02, Edinburgh, August (2002)
15. Martinoli A, Ijspeert AJ and Gambardella LM: A Probabilistic model for understanding and comparing collective aggregation mechanisms, In Floreano D, Nicoud JD and Mondada F (eds), Proc. 5th European Conference on Advances in Artificial Life (ECAL-99), Vol 1674 of LNAI, Berlin (1999) 575-584
16. McCabe TA: A Cyclomatic Complexity Measure, IEEE Trans. on Software Engineering, **2**(4) (1976)
17. Schöner G, Dose M and Engels C: Dynamics of behavior: theory and applications for autonomous robot architectures, Robotics and Autonomous Systems, **16** (1995)

18. Smithers T: On quantitative performance measures of robot architectures, *Robotics and Autonomous Systems*, **15** (1995) 107-133
19. Winfield AFT: Distributed sensing and data collection via broken ad hoc wireless connected networks of mobile robots, in Parker LE, Bekey G and Barhen J (eds) *Distributed Autonomous Robotic Systems 4*, Springer-Verlag (2000) 273-282
20. Yokobayashi Y, Collins CH, Leadbetter JR, Arnold FH and Weiss R: Evolutionary Design of Genetic Circuits and Cell-Cell Communications, *Advances in Complex Systems*, **6**(1) (2003) 37-45
21. Yourdon E: *Modern Structured Analysis*, Prentice-Hall (1989)
22. Ziemke T: On the role of Robot Simulations in Embodied Computer Science, *AISB Journal* **1**(4) (2003) 389-399