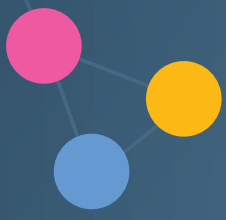


George Rzevski

Swarm AI

Meet the Next Generation of Artificial Intelligence





Springer

Swarm AI

George Rzevski

Swarm AI

Meet the Next Generation of Artificial Intelligence

Springer

George Rzevski
The Open University
London, UK

ISBN 978-3-032-26261-5 ISBN 978-3-032-26262-2 (eBook)
https://doi.org/10.1007/978-3-032-26262-2

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

If disposing of this product, please recycle the paper.

Foreword

We are living in an era where complexity is no longer the exception—it is the operating environment. The systems we depend on for health, security, transportation, energy, commerce, and public services are increasingly interconnected, fast-moving, and exposed to disruption. In such conditions, the most important question for leaders is not whether change will occur, but whether our systems can adapt with speed, integrity, and purpose.

That is why this book matters.

Much of today's public conversation about artificial intelligence is dominated by systems that generate text, images, or code. These tools have value, but they are not the whole story or the fullest potential of AI. In domains where lives, missions, and public trust are at stake, the central challenge is often not creativity—it is coordination: allocating scarce resources, responding to uncertainty, identifying emerging patterns, and making decisions that remain resilient when the situation shifts.

Swarm AI offers a compelling way to think about those challenges. Instead of relying on a single monolithic "brain", swarm approaches distribute intelligence across many interacting agents. Properly designed, such systems can be robust under stress, adaptable in the face of change, and capable of producing solutions through interaction rather than exhaustive computation. Those properties are not theoretical niceties. They map directly onto the real-world problems that public- and private-sector leaders confront every day: scheduling, logistics, contingency response, cybersecurity defence, and complex enterprise operations.

One of the strengths of this book is its discipline. It resists the urge to oversimplify or inflate. It explains core concepts in plain language while still respecting the reader's intelligence. Just as importantly, it connects ideas to

applications—showing how swarm principles can be used to build systems that continue to function when assumptions fail, when data is incomplete, or when adversaries actively seek to manipulate outcomes.

From my perspective in operational leadership and risk management, I am especially attentive to three questions whenever a new technology is proposed:

1. What problem does it solve better than existing approaches?
2. What are its limitations and failure modes?
3. What does adoption demand—technically, organisationally, and ethically?

This book equips readers to ask—and answer—those questions with clarity. It provides a framework for understanding when Swarm AI is a strong fit, when it is not, and how to evaluate claims with healthy scepticism. That balance is rare and needed, particularly as AI becomes embedded in mission systems and critical services.

Whether you are a student building your foundation, an engineer shaping systems, or a decision-maker responsible for outcomes, you will find this work both practical and thought-provoking. It does not ask you to be an expert before you begin. It asks only that you be curious—and willing to consider that the next leap in AI capability may come less from bigger models, and more from better coordination.

I hope you read this book not as a prediction, but as an invitation: to think clearly about complex systems, to design for resilience, and to pursue intelligence that serves people—reliably, responsibly, and at scale.

NATO Health Information Science and Technology Janette Arencibia,
Neuilly-sur-Seine, France

Preface

Unique Features of the Book

This book offers a clear, jargon-free introduction to Swarm AI, an approach in which many simple software agents work together to solve complex problems. Swarm AI is characterised by four practical advantages:

1. Collective Intelligence at Scale: It comprises hundreds of thousands of interacting digital agents, small decision-making programs, whose collective intelligence can produce strong group-level performance.
2. Knowledge-Driven Operation: It can be equipped with a knowledge base (explicit rules, models, and domain facts) and is not dependent on large-scale data training unlike many widely used generative AI systems.
3. Self-Organisation and Robustness: It can reconfigure itself as conditions change and is designed to improve resilience to disruption (for example, by rapidly assessing consequences and redeploying resources to minimise impact). It can also resist hacking and fraud within its environment and coevolve with the systems with which it interacts.
4. Faster Solutions Through Interactions: By enabling agents to communicate, negotiate, and coordinate, Swarm AI reduces reliance on exhaustive computational search and accelerate problem-solving in dynamic environments.

This book also surveys a wide range of commercial applications of Swarm AI, showing where it is most effective and how it can be deployed in real-world systems.

Intended Readership

This book is for readers who desire a clear understanding of the core principles of Swarm AI, an approach to artificial intelligence in which many simple software agents work together to solve complex, changing problems. It is written for:

1. General readers who want to understand how Swarm AI works, what problems it is suited for, and how it differs from more familiar approaches.
2. Decision-makers, including business executives, administrators, and public-sector leaders, who need a practical framework for evaluating where Swarm AI can help, the cost of adoption, and the organisational changes it may require.

To understand the book's content, no prior knowledge of mathematics, programming, or computer science is required. Key terms are introduced as they appear, and a glossary of key terminology is at the end of the book.

The Content

The first chapter sets out the context in which Swarm AI flourishes, namely the complexity of our geopolitical, socioeconomic, and military environments.

The second chapter explains the fundamentals of Swarm AI.

Further chapters describe some of the most important application areas: adaptive scheduling (the allocation of resources to demands), adaptive pattern recognition (the extraction of knowledge from data), adaptive semantic processing (the allocation of meaning to words), adaptive engineering (products that adapt to changes in demand), adaptive enterprise (businesses and administrations that adapt to changes in markets), and adaptive society (a society that adapts to changes rather than resists them).

London, UK George Rzevski

Acknowledgement

I am pleased to acknowledge Professor Petr Skobelev's important contribution to the development of Swarm AI. We worked together for more than 30 years and spent many hours debating how to solve bafflingly complex problems.

Competing Interests The author has no competing interests to declare that are relevant to the content of this manuscript.

Contents

1 The Twenty-First Century Is the Century of Complexity 1
The Big Picture 2
Complexity 3
Complexity and Cause-Effect Relation 5
Contrasts Between Industrial and Digital Societies 5
Complexity Is a Problem for Some and an Opportunity for Others 6
The Complexity-Tolerant Mindset 6
Only Complex Systems Can Operate Effectively in a Complex Environment 7
Only Complex Tools Can Solve Complex Problems 8
On Human Intelligence 8
Human Brain 9
On Natural Swarms 10
Do We Need Swarm AI? 10
Will AI Render Us All Jobless? 11
Can AI Dominate Humans? 11
Important Points 12
References 12

2 Fundamentals of Swarm AI 15
Swarms of Digital Agents 16
Problem Formulation for Swarms of Digital Agents 16
Self-Organisation 16
Swarms Versus Hierarchies 17

Scalability of Swarms of Digital Agents. 18
Motivating Agents by Giving Them Emotions 18
Empowering Agents to Trade by Giving Them Money 19
Asking Agents to Cost Their Transactions. 19
How Swarms of Agents Self-Improve 19
How Swarms of Agents Creatively Destruct 19
Swarm Engine 20
Knowledge Base 20
Ontology Editor. 22
Advanced Knowledge Base 22
Scene 22
A Network of Swarms. 23
An Enterprise as a Network of Swarms 24
Digital Twins 24
Example: How Swarm AI Schedules Appointments 25
Important Points 27
References 28

3 Adaptive Schedulers 29
Adaptive Order Fulfilment Schedulers 30
Adaptive Road Transportation Schedulers. 30
Adaptive Shift Schedulers 31
Adaptive Visit Schedulers 32
Case Studies 32
Adaptive Scheduler for Taxis (Minicabs) 32
The Problem 32
Previous Practice 34
The Solution 34
Results 35
Adaptive Scheduler for Delivering Cargo to the International Space Station 36
The Problem 36
Previous Practice 37
The Solution 37
Results 40
Experience 41
Important Points 41
References 41

4 Adaptive Pattern Recognition 43
Adaptive Data Clustering 43
Definitions 43
Adaptive Clustering Process 45
An Example of Clustering 46
Microeconomics of Clustering 48
An Example 49
Representing Clusters as Rules 51
Case Studies 52
Adaptive Pattern Recognition for Transportation Scheduling 52
Adaptive Pattern Recognition for Insurance 54
The Problem 54
The Solution 54
Adaptive Pattern Recognition for Travel Forecasting 55
Adaptive Pattern Recognition for Fraud and Error Detection 55
Important Points 55
References 56

5 Adaptive Semantic Processing 57
Definitions 58
The Meaning Extraction Process 58
Morphological Analysis 59
Syntactical Analysis 59
Semantic Analysis 59
Pragmatics 60
A Case Study of Adaptive Semantic Processing 61
Semantic Web 63
English to Sinhala Machine Translation 64
Important Points 64
References 65

6 Adaptive Engineering Products 67
Axial Turbo Compressors with Adaptive Geometry 67
The Problem 68
Current Practice 68
The Solution 68
Simulation 69
A Swarm of Smaller Space Robots Versus a Single Large Robot . . . 70
Adaptive Lifecycle Management of Large Aerospace Products. 71

The Problem . . . 72
Current Practice . . . 73
The Solution . . . 73
Helping with the Design of Airbus A380 Wings . . . 77
The Problem . . . 77
Current Practice . . . 77
The Solution . . . 77
Results . . . 77
Important Points . . . 78
References . . . 78

7 Adaptive Enterprise . . . 79
Introduction . . . 79
Adaptive Manufacturing . . . 79
Versatile Manufacturing . . . 80
Virtual Enterprise . . . 81
Agent-Based Model of a Virtual Enterprise . . . 82
Agent-Based Simulator of a Virtual Enterprise . . . 82
Important Points . . . 84
References . . . 85

8 Adaptive Society . . . 87
Complexity and Social Systems . . . 87
Connectivity . . . 88
Autonomy . . . 88
Emergence . . . 88
Nonequilibrium . . . 88
Nonlinearity . . . 89
Self-Organisation . . . 89
Coevolution . . . 89
Tuning Social Complexity . . . 89
Protecting Society from Extreme Global Events . . . 90
Organising Global Social Systems as a Network of Swarms . . . 90
Employing a Swarm of Agents to Detect Fraud and Errors . . . 90
Employing a Swarm of Agents to Defend Nations . . . 90
Replace Traditional Software with Swarms of Agents . . . 91
Important Features of a Complex Society . . . 91
Emergent Intelligence . . . 91
Emergent Creativity . . . 92
Emergent Leadership . . . 92

Control Versus Self-Organisation 92
Self-Organising Geopolitical System 93
Adaptive Defence 94
Adaptive Healthcare 95
Virtual NHS 95
Smart City 95
Requirements Specification 96
Evolutionary Design Method 97
Concurrent Design Mode 97
Designing Ontology 98
Designing Swarm Engine 98
Extending the Initial Design 99
Important Points 99
References 100

Swarm AI Terminology 101

About the Author

George Rzevski is Professor Emeritus, Complexity Science and AI, The Open University, Milton Keynes, UK.

He was, at various points in his career, a Visiting Professor at the Cologne University of Applied Sciences, Germany; Wuhan University, China; Samara Aerospace University, Russia; Moratuwa University, Colombo, Sri Lanka; and Brunel University, West London.

Until his retirement in 1999, George was Professor and Head of the Department of Design and Innovation and Director, Centre for the Design of Intelligent Systems at the Open University. His centre was well-funded by government and industry grants, and his department was rated 5 out of 5 in two Research Assessment Exercises.

Professor Rzevski has published widely and delivered keynote papers at numerous international conferences. In tribute to his successful research career, the Open University named his research laboratory the "George Rzevski Complexity Laboratory" upon his retirement.

At the Open University, George pioneered undergraduate education in intelligent mechatronics, launching a course in which every student experimented at home with their own intelligent robot.

His extracurricular activities included assessing candidates for tenure on behalf of several American universities, including Stanford, Ohio, and Texas, and examining over 30 PhD students from universities in the UK and abroad, including Cambridge University, Imperial College, the LSE, the Royal College of Art, Cardiff University, Singapore University, the National University of Ireland, and Moratuwa University, Colombo.

For several years, Professor Rzevski has delivered a regular series of lectures on the economic, social, and cultural implications of the Internet to post-graduate students at the London School of Economics.

He was editor-in-chief of the *Journal of Artificial Intelligence in Engineering*, published by Elsevier.

He is the author of the book, *The Future Is Digital: How Complexity and Artificial Intelligence Will Shape Our Lives and Work*, published by Springer (2023).

After retiring from the Open University in 1998, Professor Rzevski founded and managed several small businesses, through which he conducted research across broad areas of complexity science and artificial intelligence, designed AI applications, and provided consulting services in digital transformation.

His recent consulting projects include Smart City for a prosperous Swiss metropolis and Digital Insurance for one of the largest Australian Insurance companies.

George was born in 1932 in Serbia to a Russian family that had left Russia in 1917 to avoid the October Revolution.

He is an alumnus of Belgrade University and Imperial College London.

1

The Twenty-First Century Is the Century of Complexity

I write this text comfortably seated in my favourite armchair, with my laptop before me, in a green and cosy London suburb.

The sunshine floods the sitting room, and flowers smile at me from the back garden; it is an enjoyable June 2025. Idyllic, peaceful, safe?

In contrast, the headlines of *The Times*, *The Telegraph*, and *The Guardian* proclaim that the world is in deep crisis. A Third World War is most likely, if not inevitable. There are riots in Los Angeles, Northern Ireland, and England; rising unemployment in the UK; a recession in the EU; and huge national debts everywhere.

Here are some headlines.

"Will Israel's attack on Iran lead to all-out war?"
"NATO tells the UK, increase spending on defence or start learning Chinese".
"The choice between war and dishonour is fast approaching".
"It is now too late for Britain to avoid financial Armageddon".

What is going on? To answer this question, we must consider the broader context.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

G. Rzevski, *Swarm AI*, https://doi.org/10.1007/978-3-032-26262-2_1

The Big Picture

The world is transitioning from the ***industrial economy***, in which machines perform certain manual tasks more effectively than people, to the ***digital economy***, in which machines perform certain cognitive tasks more effectively than people [1, 2].

The change affects many aspects of our private lives, including banking, travel, shopping, communication, and leisure, as well as all aspects of our work across manufacturing, services, administration, and the military.

It may be helpful to remind readers that a similar socioeconomic crisis occurred about 200 years ago, triggered by the transition from an agricultural to an industrial economy.

Figure 1.1 illustrates how the complexity of our geopolitical, military, and socioeconomic environments increases as the coevolution between society and technology unfolds.

The years shown in Fig. 1.1 are 1775, when James Watt patented his steam engine, and 1989, when Tim Berners-Lee invented the World Wide Web (www). I consider these two events to mark the start of two major social changes, driven by the widespread adoption of industrial and digital technologies, respectively.

The most significant impact of the new digital technology is on communication. The widespread use of the Internet and smartphones has rapidly increased socioeconomic connectivity and, consequently, the complexity of

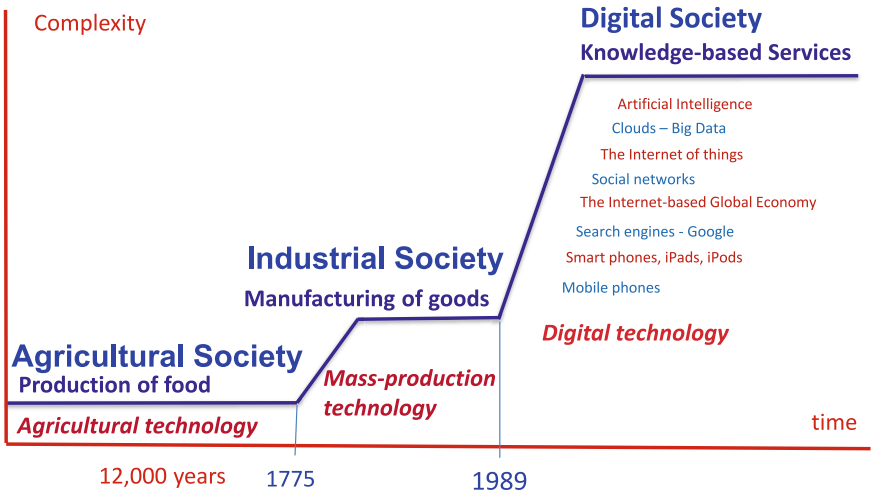


Fig. 1.1 Major steps in the coevolution of society and technology

our geopolitical, social, economic, and military environments [3]. Here are two examples.

Over 70% of the world's population, as well as nearly all businesses and governments, are connected to the Internet-based global market. They initiate, negotiate, close, modify, or cancel transactions with a click, at speeds approaching that of light, creating an unprecedentedly complex economic environment.

Self-navigating swarms of drones and cyber warfare are notable examples of the new, complex form of warfare.

Complexity

I am using the term "complex" in a scientific sense, defined as follows.

A system or situation is considered ***complex*** if it comprises many diverse, interacting participants, known as ***agents***, that operate without central control and whose behaviour is unpredictable yet not random; it follows hidden patterns [4].

Just like the Internet-based global market and the current global political, social, and military constellations.

Complexity is characterised by seven properties.

1. Connectivity

   Each agent is ***connected*** to numerous other agents.
2. Autonomy

   Agents have a degree of ***autonomy*** (freedom to choose how to behave) and are not centrally controlled. This autonomy is constrained by norms and laws, as well as by rules and regulations that vary by domain.
3. Emergence

   The global behaviour of a complex system ***emerges*** from the interactions of agents and is neither predictable nor random. It follows discernible patterns.

   To recognise these patterns, we require

   Intelligence, either natural (human) or artificial (digital), or both
   Substantial domain knowledge
   A positive attitude (beneficial, as it enables us to perceive uncertainty arising from complexity as an opportunity rather than a threat)

For those lacking these attributes, the uncertainty brought on by the current socioeconomic transition is daunting and heightens the tendency to overstate threats.

The sharp rise in uncertainty across all aspects of our lives—caused by complexity—is particularly hard to accept, given the long period of stability that preceded it.

4. Nonequilibrium

   Unpredictable, frequent disruptive events shift the operating point of a complex system ***far from equilibrium*** and occasionally to ***the edge of chaos***. Periods between two disruptive events are typically too short for a system to return to equilibrium. This makes our complex geopolitical, socioeconomic, and military environments volatile.

5. Nonlinearity

   Relations between agents are nonlinear (amplification and acceleration), which occasionally lead to extreme events. Examples are given below.

   Drift into Failure: Small errors or frauds accumulate over time and, when the tipping point is reached, cause significant damage (the accumulation of toxic loans led to a worldwide financial crisis in 2008).

   Butterfly Effect: The movement of a butterfly's wing at one end of the world may cause a storm at the other end (a single virus escaped from a laboratory and rapidly infected the whole world in 2020).

   Powerful Blow: An aggressive attempt by one agent to destroy another may cause serious disruptions worldwide (the invasion of Ukraine disrupted vital supply chains and triggered inflation and shortages in Europe, Africa, and elsewhere in 2022).

   Cloning: Copying critical software products and distributing them worldwide may cause a single software bug to trigger millions, or even billions, of disruptions (an update error in one of the most popular software products triggered simultaneous serious failures across all five continents in 2025).

6. Self-Organisation

   A complex system can change its structure to align with its environment (***adaptation***), defend itself against attacks (***resilience***), or self-improve (***innovation***).

7. Coevolution

   A complex system is ***sustainable*** and capable of irreversible change over time to remain aligned with its ever-changing environment.

Complexity and Cause-Effect Relation

It is part of human nature to seek the causes of events that affect them. In a complex environment, such a search is futile; effects may be caused by multiple interrelated factors and by events occurring in remote locations.

Contrasts Between Industrial and Digital Societies

The contrast between the Industrial Society that dominated the nineteenth and twentieth centuries and the Digital Society emerging in the twenty-first century is substantial, as illustrated in Table 1.1.

The geopolitical and socioeconomic complexity brought about by the Internet and smartphones requires fundamental change in how we conduct our private affairs and organise our business and administrative processes.

Here are some examples.

Large, hierarchically organised corporations are too rigid to operate effectively in volatile markets; networks of smaller units are more adaptable.

In the current complex geopolitical and socioeconomic conditions, globalisation is counterproductive: it accelerates extreme events such as the global financial crisis, COVID-19, and the invasion of Ukraine, which disrupt global supply chains. Transporting goods from remote factories to demand points is also energy-intensive and environmentally damaging.

Table 1.1 Contrast between the industrial and digital societies

Characteristics	Industrial society	Digital society
Key difference	Machines do certain manual work better than humans—*automation*	Machines do certain mental work better than humans—*autonomy*
Markets	Stable	Complex
Key success factor	Economy of scale	Adaptability
Key economic resource	Capital	Intelligence, natural and artificial
Key economic activity	Trading in goods	Trading in knowledge
Distribution	Motorways, railways, airways	Digital networks, the Internet
Decision-making	Centralised	Distributed
Dominant organisation	Hierarchy	Network, swarm

Nations must regain the ability to produce critical goods domestically and reduce their growing dependence on other nations, which is driven by free trade and globalisation.

Complexity Is a Problem for Some and an Opportunity for Others

To survive and thrive in a complex environment, individuals and organisations must cultivate the ability to adapt rapidly to the consequences of disruptive and extreme events.

For individuals, it means shifting from a conventional to a ***complexity-tolerant mindset***.

For businesses and administrations, this means employing powerful AI systems to replace human workers who cannot cope with the complexity, a process known as ***digital transformation***.

It is only natural that many individuals find it difficult to change their mindsets, and many enterprises struggle with digital transformation.

Excellent opportunities will be available for those who adapt first.

The Complexity-Tolerant Mindset

The complexity of our social, economic, and political environments is here to stay, at least for the foreseeable future. It is therefore prudent to learn to tolerate, even to love, complexity.

After all, if we live in a complex world, what is the point of wanting it simple?

To define the complexity-tolerant mindset, we must first consider how the world in which we live and work is constructed. The world is either designed and governed by predetermined laws or perpetually and unpredictably evolving.

Let's call these two incompatible worldviews: ***deterministic*** and ***evolutionary***.

Table 1.2 lists some important differences.

Worldview is a philosophical concept rather than a scientific one. It cannot be proved or disproved. It is a matter of belief rather than knowledge.

However, the idea that all the changes we experience throughout our lives were predetermined at the time the world was created and therefore predictable (including the invention of AI) doesn't seem realistic.

Table 1.2 Two contrasting worldviews

Salient features	Evolutionary worldview	Deterministic worldview
Creation	By Big Bang	By omnipotent force
Behaviour	Unpredictable but not random	Predictable
Changes	Due to natural and social evolution	Predetermined by design

Table 1.3 Two contrasting mindsets

Salient features	Complexity-tolerant mindset	Rigid mindset
Worldview	Evolutionary worldview	Deterministic worldview
Attitude	Complexity offers opportunities (under risk)	Complexity is a threat
Planning	Plan and replan when circumstances change	Plan and stick to your plan
Changes	Adapt to changes	Resist changes

This text reflects an evolutionary worldview.

Eminent proponents of the deterministic worldview included Aristotle, Newton, and Einstein. "God doesn't play dice with the world", Einstein declared.

In contrast, many eminent philosophers and scientists could not accept determinism, even in classical Greece. Heraclitus famously declared: "***You cannot enter the same river twice***".

Faraday was known to have doubts about determinism, and Carl Popper argued that in a predetermined world, invention would be impossible. Prigogine provided a comprehensive explanation of the evolutionary worldview in his Nobel Prize acceptance speech. It was published as a book, *Is Future Given?* [5]. S. Kaufman of the Santa Fe Institute in New Mexico published a seminal book on complexity [6].

The differences between the complexity-tolerant and the rigid mindsets are listed in Table 1.3.

Only Complex Systems Can Operate Effectively in a Complex Environment

According to the ***law of requisite complexity*** [1], to operate in a complex environment, organisations must possess requisite complexity: they must complete self-organisation between two consecutive disruptive events.

It follows that rigidly structured corporations will be unable to survive in the complex, Internet-based global market.

Only Complex Tools Can Solve Complex Problems

Complex problems are also called wicked because they change throughout the problem-solving process (due to frequent unpredictable, disruptive events).

Only adaptive problem-solving tools, such as Swarm AI, can effectively solve complex problems. Traditional software tools are too rigid.

Human intelligence is good at solving complex problems when disruptive events are infrequent. Strategic planning is a good example.

On Human Intelligence

The earliest discussion of information processing and intelligence dates back to around 400 BC. Plato thought that before using energy to move matter, one must have an ***idea*** of what, where, and how to move it.

An idea is ***information***.

And the ability to decide what, where, and how to use energy to move matter implies ***intelligence***.

At present, there is no agreed-upon definition of intelligence; however, according to the designers of the IQ test, ***pattern recognition is the primary determinant of a person's intelligence***.

For me, ***intelligence is the ability to make decisions under conditions of uncertainty*** [1].

This definition of intelligence is very comprehensive because the ability to make decisions under conditions of uncertainty encompasses many elements of intelligence, including.

Pattern recognition
Understanding the "big picture" within which the decision is being made
Information processing
Abstraction, logic, reasoning, planning, and critical thinking
Communication
Comprehending, learning, and organising knowledge into an easily accessible framework
Self-awareness

To compare Swarm AI with human intelligence and Swarm AI engines with the human brain, let's get familiar with the human thinking machine: the brain.

Human Brain

The human brain is an immensely complex system. It comprises approximately 100 billion neurons (brain cells) and around 1000 trillion synaptic connections. Each neuron may connect to up to 1000 other neurons, forming a formidably complex natural neural network [7].

It is essential to note that every brain cell contains a copy of the ***human genome***, which guides the growth, operation, and evolution of neural networks.

Humans are not born as "blank slates" (tabula rasa). Babies are born with a small, pre-wired neural network that enables them to perform basic functions, such as finding, consuming, and processing food. There is no need to train them to survive upon entering the unknown world [8].

As human offspring grow, the complexity of their neural networks increases through interaction with the environment, and their intellectual capabilities develop [9].

Scientists at University College London (UCL) have recently discovered that the two hemispheres of the human brain are wired differently [10].

The right-hand hemisphere has different synapses—connections between neurons—which enable easier creation of new network configurations and therefore facilitate ***creative behaviour***.

The left hemisphere has stronger inter-neuronal connections and is responsible for human features that rarely change, such as logical thinking and linguistic fluency.

Although some researchers dispute the notion that each brain hemisphere has a specialised function, the UCL finding that ***differences in neuronal wiring lead to different behaviours*** is significant: it helps designers of artificial neural networks achieve distinct behaviours in digital agents by configuring the wiring differently.

The importance of the human brain is evident in the resources allocated to support its functions. Although the human brain accounts for only 2% of body weight, it receives 15% of cardiac output, 20% of total body oxygen consumption, and 25% of total body glucose utilisation [11].

No single brain cell, nor a group of cells, can be described as intelligent; yet, the neural network exhibits all aspects of human intelligence. It follows that intelligence ***emerges*** from the interaction of neurons.

In other words, human intelligence is an ***emergent property*** of the human brain.

Recommended accessible reading: [12, 13].

On Natural Swarms

To understand how swarms of digital agents behave, let's look at natural swarms.

A ***swarm*** is a community of interacting members who are not centrally controlled and whose behaviour is unpredictable yet not random; it follows hidden patterns.

In other words, a swarm is a complex system.

Members of a swarm have limited autonomy—the freedom to decide how to behave—and some intelligence, enabling them to make decisions under conditions of uncertainty.

Swarms exhibit collective intelligence that exceeds the sum of their members' intelligence.

Just like a swarm of bees.

All natural ecosystems are organised as swarms—packs of wolves, herds of sheep, colonies of ants, flocks of birds, and shoals of fish. Early humans were organised in swarm-like tribes.

Natural swarms are connected in ***networks of swarms***—forests, rivers, meadows, and oceans.

In fact, Earth is a vast network of swarms that self-organise and evolve, enabling some natural ecosystems to be millions of years old.

James Lovelock first formulated the ***Gaia*** hypothesis in the 1970s, suggesting Earth functions as a superorganism, self-organising and exhibiting emergent intelligence [14, 15].

The new AI architecture borrows ideas from natural ecosystems.

Do We Need Swarm AI?

Organisations operating in complex environments, such as the Internet-based global market, must contend with frequent, unpredictable, and disruptive events that render forecasting of supply and demand impossible. As complexity increases, attempts at hacking and fraud become harder to detect.

To cope with frequent disruptions, hacking, and fraud, we need to rapidly identify the threat, determine which parts of the organisation are affected, and reorganise resources to minimise consequences without disrupting unaffected parts. To do this, we need a new, powerful tool: self-organising, adaptive AI [16], such as Swarm AI.

Will AI Render Us All Jobless?

There is compelling evidence that AI displaces humans from low-productivity jobs and creates new high-productivity employment. Examples include the following.

AI software design, coding, maintenance, and selling
AI applications design, coding, maintenance, and selling
AI hardware design, manufacturing, maintenance, and selling

Additionally, there are jobs in which people are far superior to AI. These include caring duties, dexterity, entrepreneurship, and strategy.

There will be a need for retraining courses, and I am confident that markets will do a better job than administrations of providing training in the new skills required for the introduction of AI into our lives and work. Let's rely on entrepreneurs.

Can AI Dominate Humans?

Before embarking on the design of AI applications, I spent considerable time studying human brain neurophysiology. This was time well spent. It helped me understand that we will need a significant breakthrough in the power of digital technology for AI to compete with the human brain at a strategic level, where intelligence emerges from the interaction of potentially 100 billion neurons.

It also helped me design the next generation of AI to behave as closely as possible to the human brain.

We should note that current AI products typically have hundreds, possibly thousands, and, arguably, millions of artificial neurons. But 100 billion? The difference in complexity, and therefore in the potential level of intelligence, seems unbridgeable.

Moreover, regardless of the size of the training dataset, training on data cannot match the lifelong experience of the human brain, which is shaped by visual, auditory, tactile, olfactory, and gustatory inputs. Not to mention a superior capacity for learning.

My guess is that AI will dominate the solution of highly specialised problems, but humans will remain supreme at strategic levels, in caring professions, as entrepreneurs, architects and artists, and in activities that require dexterity.

Important Points

1. The complexity of the geopolitical, socioeconomic, and military environments in which we live and work has rapidly increased.
2. To survive and thrive in a complex environment, we must adopt a complexity-tolerant mindset and develop skills in using Swarm AI.

References

1. Rzevski, G., P. Skobelev, "Managing Complexity". WIT Press, Southampton, Boston, 2014. ISBN 978-1-84564-936-4.
2. Rzevski, G., "The Future is Digital: How will Complexity and AI change our Lives and Work". Springer Nature, 2024. Available from Amazon.
3. Rzevski, G., "Complexity as the Defining Feature of the 21st Century". International Journal of Design & Nature and Ecodynamics. Volume 10, No 3 (2015), pp. 191–198. ISSN: 1755-7437.
4. Rzevski, G. "Managing Complexity: Theory and Practice". 11th System of Systems Engineering Conference (SoSE), 12–16 June 2016, Kongsberg, Norway, pp. 1–6.
5. Prigogine, Ilya, "Is Future Given?" World Scientific Publishing Co., 2003.
6. Kaufman, S., "At Home In the Universe: The Search for the Laws of Self-Organization and Complexity". Oxford University Press. 1995.
7. Edelman, G., "Bright Air. Brilliant Fire. On the Matter of the Mind". Allen Lane, the Penguin Press, London, 1992.
8. Pinker, S., "The Blank Slate: The Modern Denial of Human Nature". Allen Lane, 2002. 0713992565.
9. Posner, M. I., & Rothbart, M. K. "Educating the human brain". American Psychological Association. 2007. https://doi.org/10.1037/11519-000.
10. ScienceDaily, "Neurons Hard Wired to Tell Left from Right", 2008.
11. BrainFacts.org, How Much Energy Does the Brain Use? 2019.
12. Minsky, M., "The Society of Mind". Heinemann, 1985. ISBN 0 434 46758.

13. Noble, D., "The Music of Life". Oxford University Press, 2006.
14. Lovelock, J., "Gaia: A New Look at Life on Earth". Oxford Landmark Science.
15. Lovelock, J., "The Ages of Gaia: A Biography of Our Living Earth"
16. Rzevski, G., P. Skobelev, "Emergent Intelligence in Large Scale Multi-Agent Systems". International Journal of Education and Information Technology, Issue 2, Volume 1, pp. 64–71. 2007.

2

Fundamentals of Swarm AI

Compared with current AI products, such as Chatbox, Swarm AI offers numerous advanced features, establishing a new AI paradigm.

Self-Organisation: Capable of changing its own structure to adapt to changes in its environment, to defend itself against attacks, and to self-improve.

Coevolution: Capable of changing over time by affecting and being affected by its environment.

Scalability: Capable of working effectively with as few as three agents and as many as hundreds of thousands of agents.

Applicability: Capable of cost-effectively replacing around 50% of mental work currently performed by humans (e.g., the allocation of resources to demands).

Built-In Knowledge: Capable of working immediately; there is no need for expensive and prolonged training, depending on the quality of available data.

Learning from Humans: Capable of learning from humans when stuck, by asking for help.

Learning from Experience: Advanced versions can remove ineffective elements of domain knowledge and create new ones.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

G. Rzevski, *Swarm AI*, https://doi.org/10.1007/978-3-032-26262-2_2

Swarms of Digital Agents

Core elements of Swarm AI are ***digital agents***. *These* algorithms can compose, interpret, and exchange messages with other digital or human agents, negotiate deals, and resolve conflicts.

Digital agents have access to ***domain knowledge*** (knowledge of the part of the world relevant to their operation), which provides them with rudimentary intelligence and limited autonomy, enabling simple decisions [1–3].

Digital agents operate in swarms, a form of organisation closely analogous to a team. A ***swarm of digital agents*** exhibits all the characteristics of a natural swarm. Constituent agents cooperate or compete without central control—***swarms are complex systems***.

A swarm of digital agents exhibits ***collective intelligence*** that exceeds the sum of the individual agents' intelligence.

Problem Formulation for Swarms of Digital Agents

Problems that digital swarms solve are invariably formulated as ***resource allocation*** problems.

Which resources are allocated to which demands depends on the application.

For business processes such as order fulfilment, agents allocate ***human***, ***physical***, ***financial***, or ***knowledge resources*** to orders.
For appointments, agents allocate human resources to requests.
For shifts, agents allocate human resources to tasks.
For visits, agents allocate human, physical, and financial resources to tasks.
For pattern recognition, agents allocate data records to data clusters.
For semantic processing, agents allocate meanings to words.

This formulation, rather surprisingly, has a very broad scope. Nearly everything we do could be described as allocation, even choosing socks in the morning.

Self-Organisation

The most important ability of a swarm of agents is ***self-organisation***. It is the capacity to alter its own structure when required, without external intervention. Swarms of agents self-organise to adapt to environmental changes,

defend themselves against hackers, and coevolve with other systems with which they interact.

Here is how self-organisation works when agents allocate resources to demands.

If a new demand arrives, a new demand agent is created.
The new demand agent sends messages to the resource agents of the resources it requires, asking them to bid to fulfil the demand.
Resource agents for available resources submit their bids to the demand agent.
Resource agents committed to previous orders but dissatisfied with the arrangement may also submit bids to the new demand agent.
If the new demand agent selects only available resources, the allocation for the new demand will be completed quickly.
If the new demand agent selects some of the previously engaged resources, the conflict between the new and previous demands must be resolved before the allocation can be finalised.
To resolve conflicts, it may be necessary to compensate agents who lose matches during negotiations with virtual currency.
Negotiations for conflict resolution may take time; then, suddenly, the solution emerges.
As a result of the agent's reactions to the arrival of demand, the swarm emerges with a new structure that differs from the one it had before the demand arrived—self-organisation.

An instantaneous resolution of an issue after a prolonged period of trial and error is analogous to a clever idea suddenly appearing in a human brain and is known as a ***brainwave***.

Remarkably, a swarm of digital agents behaves like a human brain. It can even produce a brainwave.

Swarms Versus Hierarchies

Swarms have significant advantages over other organisational forms, such as hierarchies.

The ability to self-organise when a disruptive event occurs, with the aim of eliminating or at least minimising the negative consequences of the disruption without disturbing unaffected parts, makes swarms adaptive, resilient, and self-improving.

The collective intelligence of a swarm exceeds the sum of its participating agents' individual intelligences, making swarms more intelligent than monolithic artificial neural networks.

By exchanging messages between agents rather than performing computational searches, problem-solving speed increases by several orders of magnitude, making swarms considerably more cost-effective than currently popular AI approaches.

Only swarms and teams can self-organise, exhibit collective intelligence, and solve problems through communication among participants.

Scalability of Swarms of Digital Agents

Swarms work well with as few as several agents and as many as hundreds of thousands.

Hundreds of thousands of digital agents are not the upper limit. We could easily increase the number of agents to a million. However, it is prudent to partition very large, complex problems and model them as a network of swarms, just as our brain is partitioned into regions, each dedicated to a different function.

Large agent swarms are particularly effective at making rapid, consistent, and precise decisions under conditions of low to moderate uncertainty and can solve a wide range of problems arising from the complexity of our geopolitical and socioeconomic environments.

Swarms of digital agents, with access to domain knowledge, can take over intellectual tasks that humans find difficult to perform under complex conditions (e.g., rapidly responding to a hacker attack and shutting down the website).

Motivating Agents by Giving Them Emotions

Certain agents, e.g., a patient agent in a Health Service, may be assigned ***emotions***. For instance, patient agents would be ***dissatisfied*** if they waited more than a week for an appointment; ***satisfied*** if they waited no more than 3 days; and ***very satisfied*** if they obtained an appointment with their preferred clinician within 2 days.

Emotions of this kind create a propensity to initiate and sustain negotiations, regulate negotiation duration, and signal to enterprise owners the state

of their resources. A large number of dissatisfied agents indicate inadequate resources.

Empowering Agents to Trade by Giving Them Money

Money in ***virtual monetary units*** (mu) may be allocated to agents to differentiate them—for example, in a swarm where agents pay for access to other agents, those with more money can negotiate longer to increase their satisfaction.

Virtual money is also used to compensate agents who lose their previously agreed match and accept a compromise that resolves a conflict.

Asking Agents to Cost Their Transactions

To compare offers from resources to demands, for instance, which truck to use for transporting cargo, there is a need for agents to know the cost of using the resources they represent, e.g., a truck.

When a problem is solved, for example, when cargo has reached the destination, agents report the cost of the order fulfilment to the system.

How Swarms of Agents Self-Improve

If all demands are met and no new demands are arriving, agents will attempt to improve previously agreed demand-resource matches to increase swarm value. Self-improvement is initiated by dissatisfied agents. Only systems capable of self-organisation can ***self-improve***.

How Swarms of Agents Creatively Destruct

Creative destruction is a process of disruption, reform, innovation, or rebirth in which old structures are dismantled to create something new and better. The synonym is ***positive destruction***. Here is an example.

When a schedule built by agents is inadequate and cannot be improved, agents may decide to destroy it and start over.

The new scheduling process may differ from the previous one; for instance, instead of calling resource agents to compete for demands, agents may ask demand agents to compete for resources.

Swarm Engine

Software that supports agent negotiations is called a ***swarm engine*** or ***AI engine***. It is written in ***Python***.

Agents are often required to perform calculations (e.g., to decide which offer suits them better or to determine the cost of a particular activity). These calculations are handled by computational modules, known as ***microservices***. Separating calculations from agent algorithms is an important way to make agent algorithms independent and therefore usable across diverse applications.

Knowledge Base

In contrast to the current generation of AI, which is designed to be a ***tabula rasa*** (clean slate) and must be trained on domain-specific data before use, Swarm AI incorporates domain knowledge into the system and stores it in a ***knowledge base*** (KB) associated with the swarm.

Domain knowledge is divided into two categories: ***conceptual knowledge*** and ***factual knowledge*** (***data***).

Conceptual knowledge is organised as a ***semantic network*** and stored in ***ontology***, as depicted in Fig. 2.1. Nodes in a semantic network represent ***classes of domain objects*** (e.g., in the healthcare domain: clinician, nurse, paramedic, patient, ambulance vehicle, ambulance driver, and operating theatre).

Links are ***relations*** between object classes (e.g., a patient is seen by a clinician and an ambulance driver drives an ambulance) and between object classes and their ***properties*** (e.g., the properties of a clinician include qualification, specialisation, position, experience, preferences, and availability).

Data are lists of domain object instances and their property values (e.g., clinician GR has a degree in medicine from Cambridge University, is a specialist in haematology, and is a consultant with 15 years' hospital experience). Data is stored in the domain database.

A domain ontology and a domain database are two parts of the domain knowledge base.

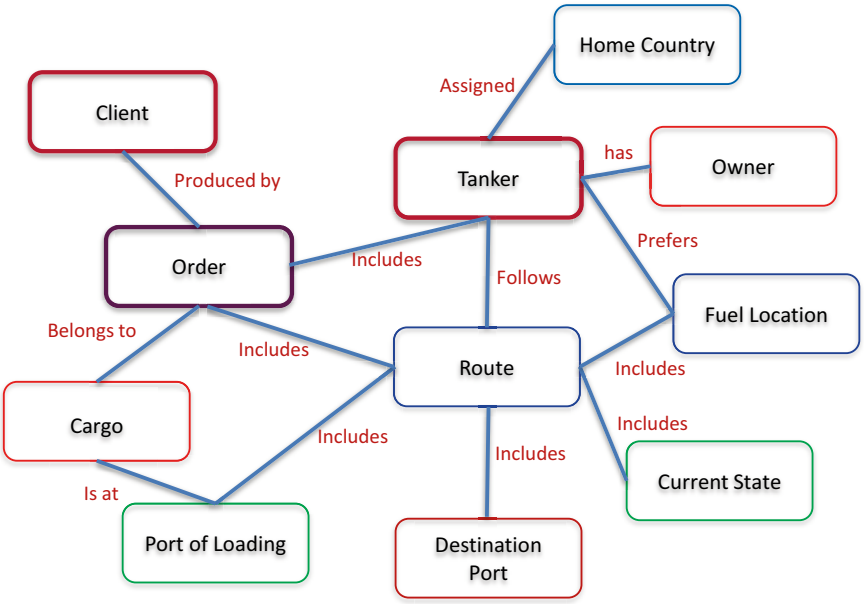


Fig. 2.1 Fragment of ontology of a scheduler for seagoing tankers

Storing domain knowledge in a separate knowledge base, distinct from the swarm of digital agents, facilitates switching between applications and significantly reduces application development costs.

Agents, when triggered into existence (for example, by the arrival of a new order), are provided with the address of a knowledge base element that contains information on how to perform the tasks for which they were created.

Examples of the knowledge collected from the knowledge base include.

How to compose, interpret, and exchange messages with other agents and, occasionally, with people—for example, by sending messages to their mobile phones.
To whom they are allowed to send messages.
How to conclude a deal.
How to resolve a conflict.
How to decide whether to cooperate or compete with other agents.
How long to continue negotiating before agreeing on a compromise.

Once they acquire this knowledge and apply it, their behaviour can be classified as intelligent with limited autonomy.

Ontology Editor

Knowledge stored in ontology can be modified by users without interrupting swarm operations. This remarkable property enables users to update the ontology whenever enterprise policies or business rules change, or when the product range is modified. A user-friendly ***ontology editor*** is provided for this purpose.

Advanced Knowledge Base

In a swarm with a basic knowledge base, agents' code must include the address of ontology where the knowledge they need is stored. This rigidly connects agents to ontology and may require additional coding if ontology needs a radical update.

A more flexible agent-knowledge relation can be achieved by attaching to ontology a ***swarm of ontology agents***, where each agent represents an object class. Then, when a swarm agent is triggered into existence, it sends messages to ontology agents to request access to knowledge relevant to its task.

Attaching a digital agent to every element of the ontology would also enable the elimination of obsolete knowledge and the transmission of messages to the ontology maintenance team, indicating the need for new knowledge, thereby enabling ***learning from humans***.

In the most advanced version of ontology, ontology agents autonomously adapt the ontology to changing domain knowledge, thereby ***learning from experience***.

All agent activities are stored and available for inspection to determine whether adjustments to the knowledge base are needed to improve agent performance. Currently, only programmers can conduct inspections.

By designing a ***swarm of evaluation agents into a knowledge base***, it is possible to continually evaluate the system's performance.

Scene

At the beginning of problem-solving, agents construct a ***scene*** (a structure representing the current state of the problem they are working on) and then maintain it by updating it whenever the situation changes [3].

For example, when a swarm of agents schedules patient appointments with general practitioners, the scene shows the current appointment schedule, which updates as appointments change.

A Network of Swarms

When the number of resources is too large for a swarm to work effectively, the allocation may be partitioned into clusters of agents, forming a ***network of swarms***, as shown in Fig. 2.2.

A network of swarms of digital agents is the new organisational structure, far more effective than the traditional hierarchy.

Its advantages are listed below.

Capacity for self-organisation.

Demand agents from one swarm may search for required resources in another swarm within the network if all resources in their native swarm are engaged (e.g., a patient agent may search for a required resource, such as a scanner, in other clinics when it is not locally available).

Reduces the danger from the butterfly effect, drifting into failure, and powerful blows.

Hierarchies and swarms of swarms, which are also hierarchies, cannot self-organise and therefore are not suitable for enterprises and individuals operating in complex markets.

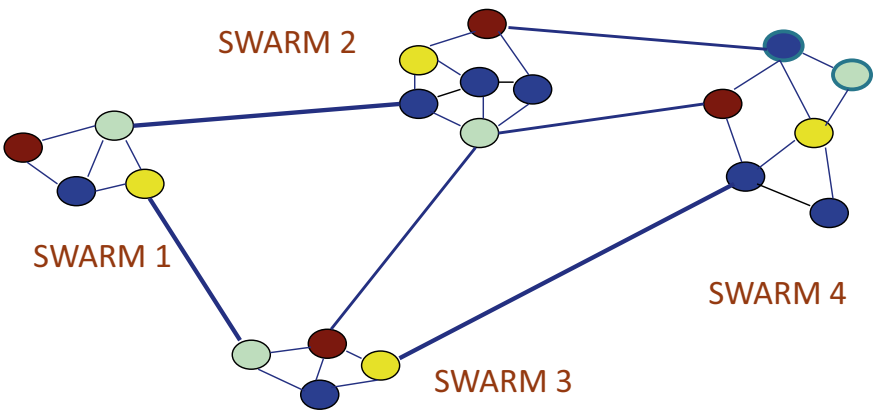


Fig. 2.2 A network of swarms

An Enterprise as a Network of Swarms

Human resources should also be organised in swarms rather than in hierarchies, to enable self-organisation, and consequently, adaptability and rapid response. However, for human swarms to function smoothly, it is necessary to ensure that malicious individuals cannot disrupt them by sending fraudulent messages to correspondents.

In a fully automated factory, robots should be organised into a network of swarms rather than hierarchies and controlled by a digital twin.

An enterprise in which human resources, robots, and digital agents are organised as swarms would exhibit emergent intelligence and could be called an ***adaptive enterprise***.

Guerrilla warfare is organised as a network of swarming fighters and is exceptionally effective at driving out invaders.

In fact, since the Napoleonic Wars, no invader has managed to win the war they started, primarily because defenders organised themselves as guerrillas.

Digital Twins

To solve a practical problem, we have to build a digital model of a relevant section of the real world, a ***digital twin***, as shown in Fig. 2.3.

Two main components of a digital twin are a swarm of digital agents representing the resources and demands of the section of the real world being modelled, and a knowledge base containing information on how that section operates.

Mathematical models, as a rule, simplify variety. Digital twins, however, represent each demand and each resource as an agent, preserving their unique properties.

The role of a digital twin is to address complex real-world problems, particularly those that are too difficult for humans to solve.

The main functions of a digital twin are as follows.

Receive and analyse information from the real world on the problem that needs to be solved (scheduling).
Solve the problem (create a schedule).
Send the solution (a schedule) to the real world.
Repeat the cycle whenever there is a change in requirements (rapidly reschedule and send the updated schedule).

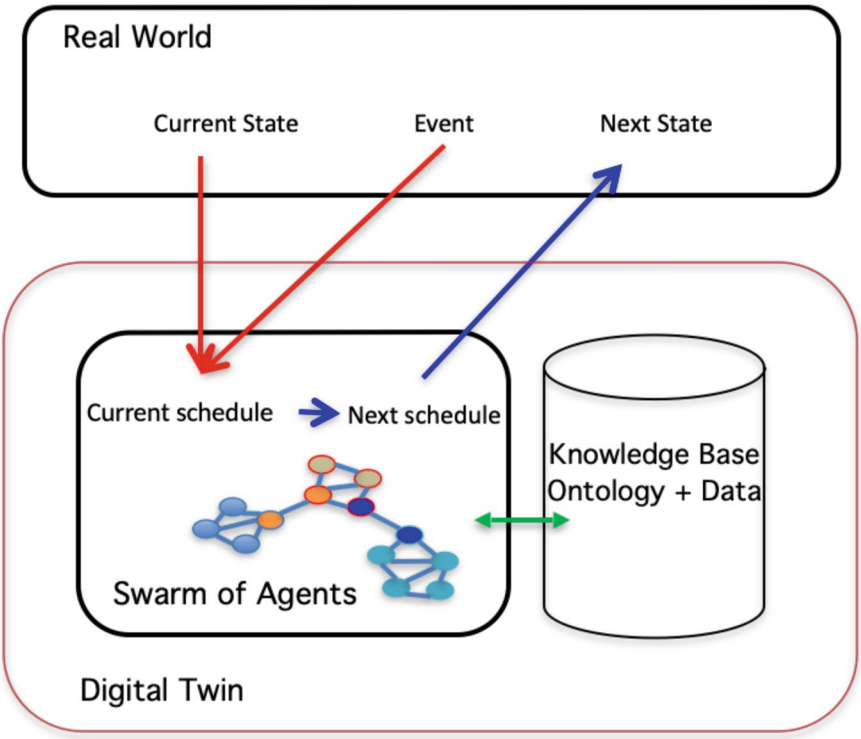


Fig. 2.3 Digital twin

Example: How Swarm AI Schedules Appointments

We illustrate how a swarm of digital agents solves the practical problem of scheduling a patient's surgery appointment with three general practitioners (GPs), two nurses, and a blood-testing laboratory.

Human resources are GPs and nurses.
The physical resource is the blood-test laboratory.
Demands are patients' requests for appointments and doctors' requests for blood tests.

The first step is to build a digital twin of the surgery: a swarm of digital agents representing surgical resources and demands, and a knowledge base containing information on how this particular surgery schedules appointments (see Fig. 2.4).

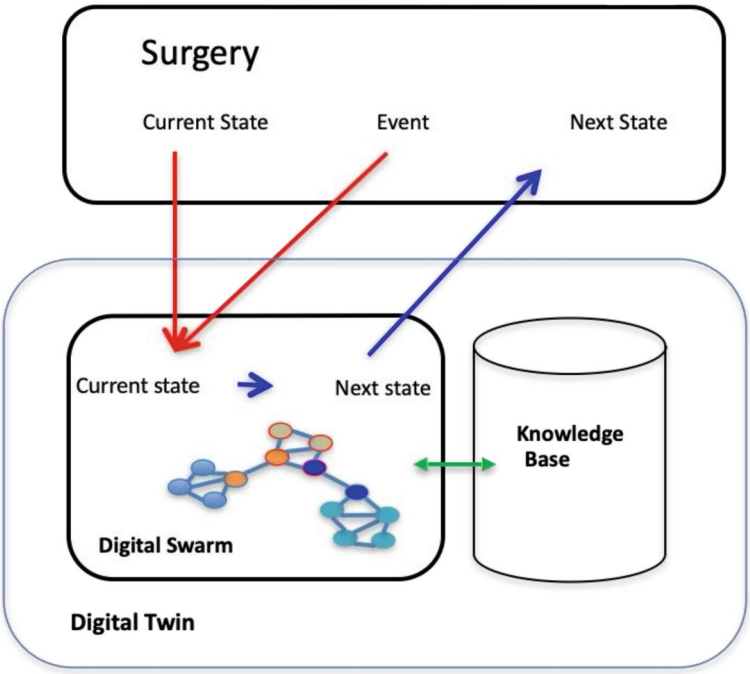


Fig. 2.4 Surgery and its digital twin

Here is how the swarm of digital agents makes appointments.

When the first patient, Patient1, calls the surgery to request an appointment with a GP, a Patient1 Agent is created.
Patient1's Agent consults the knowledge base to determine procedures for scheduling an appointment for Patient1.
Patient1 Agent sends messages to GP1 Agent, GP2 Agent, and GP3 Agent, enquiring about their availability.
GP Agents respond, stating when the GPs they represent are available.
The Patient1 Agent selects the most convenient appointment and sends confirmation messages to all relevant agents and to the patient's smartphone. Assume the appointment is scheduled for Patient1 to see GP2.
If the GP2 runs late, his agent sends messages to other GP agents, enquiring whether they can see the patient.

If GP1 is unexpectedly free earlier, his agent will offer his services to the Patient1 Agent.

Patient1 Agent agrees and sends confirmations to all concerned.

Suppose GP1, upon examining the patient, decides that a blood test is required. In that case, his agent will negotiate with the Nurse1 Agent, Nurse2 Agent, and Lab Agent regarding availability and will schedule an appointment while the patient is still in surgery.

The Patient1 Agent sends a text to the patient's smartphone asking for confirmation.

If two patients prefer the same general practitioner and request appointments for the same day and time, the conflict must be resolved before the allocation can be agreed. Conflict resolution is achieved through negotiation between the relevant agents, with input from the patients.

When the appointment is complete, each resource agent asks the accounting module to calculate the cost of its services to the patient, and then sends a report to the surgery system.

For agents with emotions, we could specify that the appointment is finalised when the Patient Agent is "fully satisfied," Meaning, for instance, that the patient got an appointment with his first-choice general practitioner within one day of the request.

Important Points

1. The widespread use of the Internet and mobile phones has increased connectivity and, therefore, the complexity of the global market.
2. To prosper under conditions of complexity, organisations and individuals must be adaptive, capable of rapidly rescheduling resources in response to unpredictable, disruptive events. This can be achieved only by swarms of digital agents capable of self-organisation and exhibiting collective intelligence greater than the sum of the intelligence of the constituent agents.
3. Swarms of digital agents are ideally suited for work in a complex environment. They can quickly identify a disruption and minimise its adverse effects.

References

1. Rzevski, G., P. Skobelev, "Managing Complexity". WIT Press, Southampton, Boston, 2014. ISBN 978-1-84564-936-4.
2. Rzevski, G.; P. Skobelev; A. Zhilyaev "Emergent Intelligence in Smart Ecosystems: Conflict Resolution by Reaching Consensus in Resource Management". Mathematics, 10, 1923. 2022.
3. Rzevski, G., "The Future is Digital: How will Complexity and AI change our Lives and Work". Springer Nature, 2024.

3

Adaptive Schedulers

A scheduler is adaptive if it can rapidly detect unpredictable, disruptive events, identify affected parts, and reschedule them to eliminate or at least minimise the negative consequences of the disruption without disturbing unaffected parts [1].

To understand how adaptive schedulers work, we need to have a fresh look at the concept of organisation.

An organisation can be considered a collection of resources assembled to meet specific demands.

Human resources—managers, employees, and external consultants
Physical resources—buildings, furniture, land, machines, trucks, and computers
Financial resources—working capital and investments
Knowledge resources—policies, rules, and instructions on how to operate business processes

Then, the basic operation of an organisation is the allocation of available resources to individual demands as they arise.

The allocation of human, physical, and financial resources to orders (drivers, trucks, routes, and budgets to transportation orders)
The allocation of human, physical, and financial resources to customer visits (maintenance engineers, tools, and budgets)
The allocation of human resources to shifts (clinicians to clinics)

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

G. Rzevski, *Swarm AI*, https://doi.org/10.1007/978-3-032-26262-2_3

The allocation of human resources to appointment requests (general practitioners to patients)
The allocation of knowledge resources to human resources (training courses to students) requests.
The allocation of financial resources to organisational units (working capital to departments)

Current batch schedulers and optimisers are not suitable for operating in a complex market characterised by frequent, unpredictable, disruptive events. Only adaptive schedulers can effectively allocate resources to demands in the face of market complexity.

Adaptive Order Fulfilment Schedulers

To autonomously fulfil orders received by an enterprise, we build a digital twin of the enterprise—a swarm of digital agents representing orders and resources. Knowledge of order fulfilment for this particular enterprise is stored in the twins' knowledge base.

Through negotiations between the Order Agents and Resource Agents, appropriate enterprise resources are assigned to received orders, and the order-fulfilment business process is established.

When a disruptive event occurs, resource allocation to orders is rescheduled to maintain a balance between the satisfactory fulfilment of individual orders and the maximisation of the value of the overall order fulfilment process.

Adaptive Road Transportation Schedulers

The order fulfilment process for a road transport operation may include the following classes of objects: transportation order, truck, driver, loading/unloading bay, loader, and cargo.

A swarm for allocating road transport resources to orders may include the following agents: truck agents, driver agents, bay agents, loader agents, cargo agents, and order agents.

Initially, agents construct a scene—the current state of the order fulfilment process—a document that depicts all enterprise resources: tracks, drivers, bays, and cargoes and their roles in fulfilling accepted orders. For instance, truck 32 transports cargo 56 for order F to unloading bay S21.

When a transportation order arrives, the order agent is created.

The new order agent sends messages to the appropriate truck agents, inviting them to bid on transporting the cargo to its destination.

Truck agents available for order fulfilment perform the following tasks:

Ask a routing service to create the best route to the destination.
Ask a booking service to book loading and unloading slots with the bay agents.
Ask an accounting service to calculate order fulfilment expenses and compose a bid, including profit.
Send their transportation bids to the order agent.
The order agent chooses the most cost-effective bid and informs the selected truck agents.

Agents for selected trucks send messages to driver agents, inviting them to bid to drive those trucks.

Agents of selected bays request bids from loader agents for loading and unloading.

All resource agents request that the accounting service calculates the costs of their transactions.

The scene is updated to reflect the changes resulting from the acceptance of the new order. The order fulfilment schedule is created and sent to all concerned.

The fulfilment of each order is monitored using GPS on trucks and messaging. If a disruptive event occurs (truck failure, driver no-show, delay, or order cancellation), the schedule is appropriately modified in real time through negotiation among the affected agents.

Adaptive Shift Schedulers

A shift is a specific work period or a designated task that needs to be completed. It has a defined start and end time, along with any breaks. For example, a restaurant might have day, evening, and night shifts.

Shift allocation is determined by enterprise-specific rules. Object classes may be: shift, human resource, and physical resource.

A swarm for allocating resources to shifts may include the following agents: Shift Agents, Human Resource Agents, and Physical Resource Agents.

The first step is to construct the initial scene that depicts how resources are currently allocated to shifts.

Agents representing shifts with insufficient resources send messages to relevant resource agents inviting them to submit bids.

Resource agents who received the invitation request that the accounting service calculates the costs of using the resource they represent and compose bids.

Upon receiving the bids, the shift agents select the best offers and send acceptance messages to selected resource agents.

Selected human resource agents send messages to smart smartphones of the human resources they represent, and upon receiving answers, communicate them to the shift agents.

The agreed schedule remains valid until a disruptive event occurs, such as a new demand (e.g., a large group booking) or a no-show by a human resource.

Adaptive Visit Schedulers

Visits for equipment sales, maintenance, repair, and installation, as well as attendance at courses, meetings, or conferences, are expensive and time-consuming activities that must be carefully scheduled.

Classes of objects may include visitors, locations, tasks, and tools required for those tasks.

The agents may be a visitor agent, a location agent, a task agent, and a tool agent.

If we assume that visitors are repair technicians and tasks are appliance repairs, the allocation process would begin by task agents inviting bids from repair technician agents.

Case Studies

The case studies in this section are brief descriptions of some of the commercial adaptive schedulers our team has designed and delivered to clients.

Adaptive Scheduler for Taxis (Minicabs)

The Problem

The taxi service for which the adaptive scheduler has been designed had the following characteristics:

A modern ERP system and a call centre with about 130 operators receiving orders concurrently; some orders were received through the company website.

A large fleet of more than 2000 vehicles, each with a GPS navigation system.

More than 13,000 orders per day. The order flow occasionally exceeded 1500 orders per hour. Order arrival times and locations are unpredictable.

The order attributes were: place of pick-up and drop-off; urgent or booked in advance (for a specific date and time); type of service (minivan, VIP, etc.); importance (a number from 0 to 100, depending on the client); and special requirements (pet, need for a child seat, etc.).

A large variety of clients: personal, corporate, VIPs, with a variety of discounted tariffs, with special requirements for drivers, disabled, requiring child seats, requiring transportation of pets, etc.

A wide variety of vehicles, including minivans and cabs, some equipped with specialised features to meet clients' specific requirements.

A large number of freelance drivers leased cars from the company and were permitted to start and finish their shifts at times of their choosing, which may vary from day to day.

At any given time, around 700 drivers were working concurrently, competing for clients.

Vehicles and drivers had the following attributes: vehicle type; capability to complete special jobs; driver experience (novice or experienced); place of residence; current vehicle location (GPS coordinates); driver status ("unavailable", "break", "working", "free", "will be free in 5/10 minutes", and "goes home").

Guaranteed pickup of clients in the centre of London within 15 min from the time of placing an order.

The unpredictability of traffic congestion across various parts of London causes delays and, consequently, schedule disruptions.

Unpredictability of times spent in queues at airports and railway stations.

Disruptive events included: the arrival of new orders; changes or cancellations to orders; changes to driver profiles (e.g., vehicle type); changes to driver status or location; client no-shows; and vehicle failures.

Several exceptions to the general requirement to find the best economic match between a vehicle and a client include: matching drivers who drive home after finishing their shifts with passengers travelling in the same direction (to reduce drivers' idle runs) and giving priority to drivers who, on a particular day, have had less work than others (to increase drivers' satisfaction with working conditions). These exceptions may be changed at any time.

Scheduling vehicles and drivers under such conditions is an exceedingly complex process that is not feasible with any known mathematical method.

Previous Practice

Before the adaptive scheduler was implemented, the company used manual scheduling by a large number of very skilled dispatchers.

The Solution

The adaptive scheduler, designed for this taxi service, works as follows [2]:

To achieve a solution as close as possible to the optimum within a short interval between two disruptive events, the system introduces a delay between allocating a vehicle to an order and sending the pickup instructions to the driver.

During this time interval, the vehicle is considered available for new allocations, provided that, despite the required rescheduling, the reallocation will improve performance indicators. During this process, vehicle agents attempt to reach an agreement with one another, taking into account previously allocated orders and allowing them to be rescheduled. The rescheduling of allocated resources triggers a wave of negotiations to resolve conflicts between new and existing orders. The length of the rescheduling chain is limited only by the time required to reach a client in the busy city of London, which is normally sufficient for several schedule changes.

The system builds a schedule and continually reviews it as long as there is a possibility to improve key performance indicators and decision-making time remains available.

Order agents and resource agents compete with one another or cooperate, depending on the requirements. In the case described above, it can be inferred that an agent sacrifices its allocation to create an opportunity for another agent to obtain an allocation, thereby improving the overall performance of the service.

System functions in cycles during which it collects disruptive events and enqueues them. In each cycle, the events in the queue are processed one by one, and agents are, in turn, granted control by the dispatcher. Each event initiates a chain of negotiations between agents. Once all events have been processed, the system enters a sleep state until a new event arrives.

Scheduling workflow includes the following steps:

A new order comes to the event queue.
The possibility of scheduling orders is assessed.
The order receives an agent.
All drivers who can complete this order are included in the pre-matching process.
Evaluation of all driver-order pairs is based on microeconomic criteria.
The order agent requests order completion costs from selected driver agents. This cost includes the transfer of the order from the previously assigned driver. The driver agent receives information about reallocation costs by sending a request to its current order agent.
If the revised decision is better than the previous one, it is applied.
The process continues for all candidate drivers, for whom the initial evaluation (without transfers) was better than the current evaluation.
If no changes occur during the cycle, the disruptive event processing is considered finished.
The decision-making time for each order is dynamically calculated. The introduction of dynamic decision-making time increased fleet effectiveness by reducing the average task completion time per driver.
When the decision-making time expires, information about the order is sent to a driver, and the order is removed from the scheduler.

To the best of my knowledge, this is the first scheduler that allocates taxis to orders in full operation.

Results

Results were exceptionally good.

The system was commissioned 6 months after the project began.
The time required to repay investments was 2 months from the beginning of the operational phase.
During the first month of operation, fleet utilisation effectiveness increased by 5–7%, which, in absolute terms, enabled the fleet to complete additional orders, resulting in 5 million dollars in annual revenue.
98.5% of all orders were allocated automatically without the dispatcher's assistance.
The number of lost orders was reduced to 3.5 (by up to 2%).
The number of vehicle idle runs was reduced by 22.5%.

Each vehicle completed two additional orders per week, spending the same amount of time and consuming the same amount of fuel, increasing each vehicle's yield by 5–7%.
The additional income was distributed between the company and the drivers. Driver wages have increased by 9%.
Further improvements included reducing delayed pickups by three times, thereby significantly improving service levels for customers.
The average urgent order response time (from booking to pickup arrival) was reduced to 9 min, the shortest in London.
For high-priority orders, the response time was no more than 5–7 min. A reduction in response time was especially noticeable in overloaded areas.
Search for orders on the way home, and an improved optimisation mechanism, compared with a previous system, resulted in a reduction of 3–4 thousand miles in the daily fleet run, which was beneficial to both drivers and the city's ecology.

Adaptive Scheduler for Delivering Cargo to the International Space Station

The Problem

Logistics for the International Space Station (ISS) is a complex business process consisting of many interrelated processes and tasks, including:

Scheduling spaceflights (with distinct operations of launch, docking, and undocking).
Scheduling delivery of fuel, spare parts, and extravehicular activity (EVA) gear.
Supporting ISS crew life by scheduling the deliveries of food, water, and healthcare items, and the collection of waste.
Supporting ISS scientific experiments by scheduling the deliveries of laboratory equipment, materials, and instruments, and the return of experimental results to Earth.
Allocating delivered supplies to storage zones within ISS.
Ordering of required resources from suppliers.
Managing projects on the design and development of resources that will be produced in-house.
Scheduling production of in-house developed resources.

All activities are conducted under severe space, weight, and time constraints, and amid frequent disruptive events.

Even a small change in a schedule triggers a ripple effect across other schedules, thereby disrupting all linked systems.

A large number of stakeholders must be involved in decision-making, thereby increasing the problem's complexity.

Previous Practice

Resource allocation to demand was previously performed manually by large numbers of scientists, engineers, and managers, who had to run thousands of iterations and interactions to find a compromise solution using simple tools such as spreadsheets.

The Solution

The problem was deemed exceedingly complex, and the decision was made to design several adaptive schedulers and connect them into a network [3].

The network configuration was preferred because it enabled easy expansion. Additional schedulers could be connected at a later stage.

To the best of my knowledge, this is the first fully operational network of adaptive schedulers worldwide.

The following adaptive schedulers were initially designed and connected to the network:

1. Adaptive scheduler for the flight program—launching, docking, and undocking of spacecraft to ISS modules.
2. Adaptive scheduler for the deliveries of cargo, including fuel and water, and the return of waste.
3. Adaptive scheduler for the cargo storage allocation.
4. Adaptive scheduler for the delivery of equipment for scientific experiments.
5. Adaptive scheduler for resource management, including in-house software and hardware production and procurement.

All ISS logistics domain knowledge is captured in ontology and factual databases, which together constitute the ISS logistics Knowledge Base.

Let's consider, for example, the flight program ontology.

Domain object classes include: ISS, ISS Module, Flight, Spacecraft, Spacecraft Engine, Expedition, Crew Member, Cargo Unit, Fuel, Port, Flight Program, Schedule, etc.

Object classes are connected by relations into a network that can be modified by users via a user interface, allowing ontology to be edited without system shutdown users can, if necessary, introduce new types of spacecraft, specify which ISS ports they can dock at, new types of ISS modules, and new types of cargo and flight operations. The user interface for the interactive flight program scheduler will adapt accordingly, providing new capabilities.

The relationships between ontology object classes and their attributes describe the preferences and constraints that agents use in their negotiations with one another. For example, the relations indicate which spacecraft types can be used for waste disposal; their maximum payload; the time intervals between launch and docking; spacecraft preferences for ISS docking ports and cargo units; ISS port preferences for different spacecraft; the required emergency reserve on board the ISS; and the minimum payload of a specific cargo type required on each flight.

In addition to generic agent types, such as order and resource agents, ISS schedulers employ specialised agent types. For example, the flight-program scheduler uses flight, ISS port, and docking agent types, whereas the cargo-flow scheduler uses cargo and flight agent types. Some agent types appear in multiple swarms.

To facilitate interaction between schedulers, every pair of swarms shares at least one agent. For example, the flight agent is shared between the flight-program scheduler and the cargo-flow scheduler.

If the spacecraft launch date is postponed due to a preparation delay, the agent for the delayed flight, resident in the flight-program scheduler, will update its status, i.e., change the launch, docking, and undocking dates. Because it is active in both the flight-program and cargo-flow schedulers, its message about the change will alert cargo agents in the cargo-flow scheduler to the delay in their flight and allow them to negotiate a transfer to another flight, if necessary.

Conversely, if some cargo deliveries are reduced, the flight's capacity can become underutilised, which will be reflected in the cargo-flow scheduler as a "partially satisfied" flight agent. As a result, this agent will seek to increase its satisfaction by recruiting new cargo agents to join its flight.

Here is an example of a typical agent interaction.

When a new request to allocate a cargo unit to a certain flight arrives from an ISS system curator, a new cargo agent is created.

The new cargo agent interacts with flight agents with launch dates in the current flight program to search for an allocation opportunity. This interaction involves sending messages between the cargo-flow scheduler and the flight-program scheduler.

Available options are prioritised based on cargo requirements and transaction values.

If there is sufficient free space on the selected flight and the cargo fits within the flight's weight and size limits, the new delivery is scheduled.

Alternatively, the following actions are performed:

- The new cargo is added to the flight's current payload, causing an overload.
- The flight agent requests that cargo agents reduce the flight's payload. Cargo agents are ranked by priority and preferred delivery date, and the lowest-ranking agents are removed from the flight.
- If the current payload still exceeds the maximum capacity, the previous step is repeated until requirements are met.
- Cargo agents removed from their preferred flight attempt to reassign themselves to the next available flight, potentially displacing other cargo agents.
- If the cargo agent is unable to assign itself to any flight in the current flight program, its priority is deemed too low, and it will remain unassigned until the next change in the flight program, such as moving a cargo spacecraft launch date from January of the next year into December of the current one.

Two points here are of high importance.

Not all cargo units are independent; two or more cargo units may be linked as parts of the same set or placed in the same box. In such a case, before a cargo unit is moved to another flight, the system considers the interests of other linked cargo units and the overall profitability of the move.

When one or more cargo units move, they may create free space in excess of the amount required to accommodate a new cargo unit. In this case, it is necessary to reschedule oxygen, water, and/or fuel. If oxygen, fuel, and/or water were previously reduced in volume, they now have the opportunity to restore them. In the negotiation process described above, it is impossible to predefine a sequence of steps or logic. The negotiation process is not an algorithm. It adapts to its environment.

The user interface is highly adaptive and customisable, enabling each user to personalise their workspace and reduce work-related stress.

Results

The total time saved annually varies across scheduling areas from 30 to 540 man-hours. The saved time was used for research and comparison of flight program options, as well as for analysing feasible responses to disruptive events.

However, the exceptional speed of rescheduling in emergencies proved to be life-saving when two disasters struck the rocket-launching company.

In 2015, for the first time in spaceflight history, two consecutive cargo deliveries to the ISS failed. The Russian Progress M-27M vehicle was lost on April 28th, followed by SpaceX CRS-7 on June 28th.

Due to the failures, it was necessary to reschedule all flights and deliveries quickly.

The speed at which our adaptive scheduler completed this task was truly remarkable. As a result, the consequences of failures were not disastrous.

The next manned flight was postponed by 2 months to allow time to determine the causes of the failures. To ensure continued human presence aboard the station, the departure of the docked Soyuz TMA-15M had to be delayed by a month, inadvertently setting records for the longest continuous female spaceflight and for the European Space Agency (ESA) astronaut spaceflight.

Although the astronauts on board the ISS had no shortage of supplies due to the properly determined emergency reserve, the depleted stock had to be replenished; therefore, the next Progress cargo delivery mission had to be moved forward by 1 month. To maintain the necessary number of cargo deliveries after the loss of one vehicle, the launch of Progress M-29M had to be moved from early 2016 to late 2015.

The cargo payload for the next resupply mission required a complete rework. Some of the lost 2500 kg of cargo could be readily replaced (oxygen tanks, food, and medical equipment), but the quantities had to be rescheduled to accommodate both the departing crew's longer stay and the next-arriving crew's shorter stay and to replenish depleted emergency stocks. Cargo units deemed least important (e.g., tissue rolls, spare manuals, and trash bags) had to be excluded from the cargo flow entirely without affecting the ISS crew. Several of the lost cargo units, which were critical and existed only as unique specimens (e.g., air absorption filter shells and electrical current converter batteries), had to be rescheduled for later flights until replacements could be manufactured. The loss of SpaceX CRS-7 also led to the cancellation of several EVAs, reducing the need to replace several single-use components.

Finally, the scientific equipment for multiple experiments conducted by each space agency was lost following the two launch failures. As with other

cargo, some equipment could be replaced in the next supply delivery, whereas other items required a lengthy manufacturing process, thereby affecting the experimental schedule. As a consequence, several experiments scheduled for later dates had to be brought forward to ensure the flight crew's maximum effectiveness during their extended stay.

Experience

Over the past 20 years, the author's team has designed and commissioned numerous adaptive schedulers for commercial clients. In addition to the London taxi scheduler and the network of ISS schedulers, our team has designed and commissioned adaptive schedulers for seagoing crude oil tankers, car rentals, soft drink distribution, online shopping deliveries, repair engineers' visits, road transport, and manufacturing [4].

Several systems, designed in 2000, remain operational, giving their clients a considerable competitive advantage.

Perhaps the most complex and prestigious accomplishment is the adaptive scheduler that delivers crew and cargo to the International Space Station.

Important Points

1. Rapid increase in market complexity renders batch scheduling, such as ERP, obsolete. It is expected that next-generation agent-based adaptive schedulers will be in high demand.
2. Two case studies describe how Swarm AI schedulers have solved highly complex problems of scheduling taxis and spacecraft.

References

1. Rzevski, G., P. Skobelev, Agent Method and Computer System for Negotiating in a Virtual Environment. Patent Reference No: WO 03/067432 A1. Published 14.08.2003.
2. Glaschenko, A., Ivaschenko, A., Rzevski, G., Skobelev, P. "Multi-Agent Real Time Scheduling System for Taxi Companies". Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009), Decker, Sichman, Sierra, and Castelfranchi (eds.), May 10–15, 2009, Budapest, Hungary. ISBN: 978-0-9817381-6-1, pp. 29–35.

3. Rzevski, G., Soloviev, V., Skobelev, P. & Lakhin, O., "Complex Adaptive Logistics for the International Space Station", International Journal of Design & Nature and Ecodynamics, Volume 11 No. 3 (2016), pp. 459–472. ISSN: 1755-7437.
4. Rzevski, G., "The Future is Digital: How will Complexity and AI change our Lives and Work". Springer Nature, 2024.



4

Adaptive Pattern Recognition

Pattern recognition is adaptive if it can recognise patterns that change unpredictably during recognition.

Due to the worldwide use of the Internet and smartphones, market volatility has increased to such a level that it is difficult, if not impossible, to predict many of the most vital business aspects—demand, supply, investment availability, and cash flow.

At the same time, we are facing a rapid rise in cybercrime and need new detection and protection methods.

And perhaps most importantly, the complexity of the geopolitical environment and the development of new forms of warfare, including autonomous drones and cyber warfare, compel us to focus on novel methods of defence.

We can summarise the current state of the planet by stating that the world is complex, with many key aspects of life unpredictable but not random.

With the right tools, we could recognise patterns of critical behaviours in markets, crime, warfare, and elsewhere through adaptive data clustering.

Adaptive Data Clustering

Definitions

Data clusters are groups of records (data elements) that share common features, such as records of all customers who purchase bread and milk daily. Such a data cluster illustrates a pattern of customer behaviour and can be employed to manage customer relations.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

G. Rzevski, *Swarm AI*, https://doi.org/10.1007/978-3-032-26262-2_4

Data clustering is the process of grouping records into clusters based on a defined set of clustering criteria.

Adaptive data clustering is a clustering process in which the dataset to be clustered changes unpredictably.

Clustering propensity is the extent to which record agents proactively search for clustering partners. To achieve clustering propensity, each data element is assigned an energy level.

The energy level of a record or cluster, measured in agreed-upon ***energy units*** (eu), determines its ability to search for the optimal clustering configuration and, consequently, its ability to influence the process of changing clustering configurations. The energy level may also be used to limit the time required to achieve an acceptable clustering solution.

Energy levels could be distributed evenly across records or according to domain-dependent rules. For example, in e-commerce applications, the energy level available to a record can be set as a commission for each item sold. Then the record of a sale of a batch of goods would have a higher energy level than the record of a sale of one item.

In general, records that are more important to the user are assigned higher energy levels. This works as follows. Records spend their energy to join, form, or leave clusters. Important records with higher energy can participate in more clustering negotiations to achieve optimal clustering membership. In contrast, less important records are limited to fewer searches. Clusters that attract more important records accumulate higher energy levels and are therefore more visible to users.

Clustering criteria determine how records are clustered.

Cluster density is the number of records contained in a cluster.

Cluster valuation formulae specify how cluster agents assess a cluster's value. Generally, a cluster's value depends on its density, the energy levels of its members, its shape (or boundaries), the number of attributes associated with the cluster, the variety of those attributes, and other factors. A simple and effective way to determine a cluster's value is to equate it with the cluster's density.

Record valuation formulae specify how cluster agents are to assess a record's value. If cluster density were maximised, records that increase it would be preferred.

System value is the value of the overall clustering process. The guiding principle for allocating records to clusters is to maximise the system value.

Adaptive Clustering Process

Let us assume that a digital swarm is tasked with allocating records to clusters, and that those records arrive in the system in small batches. The arrival times of records and their features are unpredictable. Then, the real-time clustering method is as follows:

An agent is assigned to a new record upon its arrival in the system.
The new Record Agent considers available clusters, selects those that appear attractive (as determined by a cluster valuation formula), and submits membership applications to the appropriate Cluster Agents.
Cluster Agents receive membership applications and evaluate applicants using a record valuation formula. Those Cluster Agents who decide that the new applicant will increase the value of their clusters send membership offers to the applicants.
The Record Agent accepts the most appropriate offer and joins a cluster.
If no suitable cluster is available, the Record Agent attempts to form a new cluster with other records, whether or not they belong to clusters, by sending cluster formation proposals to their agents.
The Record Agents, to which the formation of new clusters is proposed, consider the offer. They accept the offer only if it increases the overall value of the system.
By accepting the offer, agents effectively reorganise the whole system—the previously established relationships between the released records and their clusters are destroyed, and new relationships between different records are established, thereby increasing the overall value of the system—a process previously defined as self-organisation.
Agents representing newly created clusters and/or clusters whose properties (value, boundaries, and density) have changed during the self-organisation start a new negotiation round with agents of selected records, repeating the process described above.

The clustering process continues until all records are assigned to clusters and no further change in cluster membership increases the value of the system, or until the available time for clustering is exhausted.

Under conditions of perpetual arrival of new records to the system, at some point in the clustering process, agents will begin dropping out-of-date records from further clustering considerations—the process known as ***evolution***.

The real-time data clustering using swarms of digital agents is protected by patents [1, 2].

An Example of Clustering

Four records (data elements) arrive in the system one by one: Record 1, Record 2, Record 3, and Record 4 (Fig. 4.1).

The cluster valuation formula is based on the density of clusters, and the negotiation rule is "first consider the nearest data element or cluster".

Then the clustering steps will be as follows:

Record 1 arrives at the system.

Record 2 arrives at the system. It forms with Record 1 a new cluster, Cluster 5 (Fig. 4.2).

Record 3 arrives in the system. It applies to Cluster 5 for membership, but it is rejected because its membership would reduce the cluster density.

Record 3 then suggests to Cluster 5 to form a new cluster, which would include Record 3 and Cluster 5.

They agree and form Cluster 6 (Fig. 4.3).

Record 4 arrives in the system.

Record 4 suggests that Record 3 leaves Cluster 6 and joins Record 4 in a new cluster.

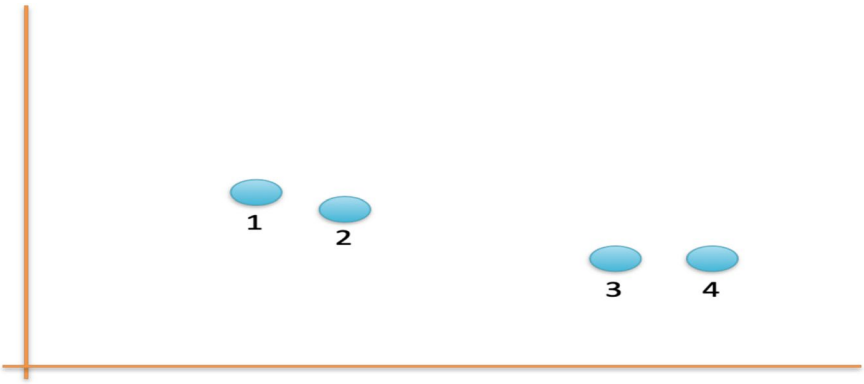


Fig. 4.1 Records available for clustering, arriving one by one

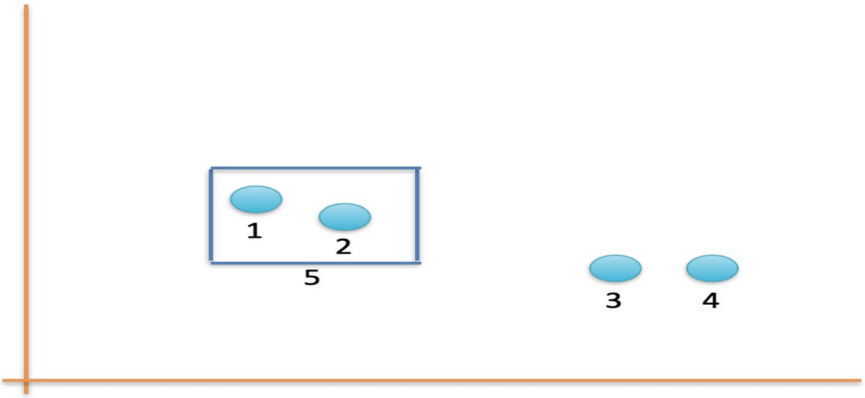


Fig. 4.2 Records 1 and 2 form Cluster 5

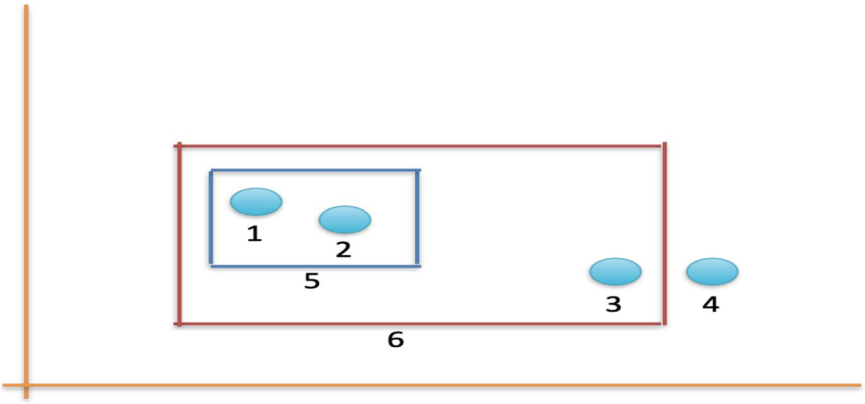


Fig. 4.3 Record 3 and Cluster 5 form Cluster 6

Record 3 agrees because the new cluster would have a greater density than Cluster 6. Cluster 6 is destroyed, and Cluster 7 is created from Records 3 and 4 (Fig. 4.4).

Cluster 7 then proposes to Cluster 5 to form a new cluster. They form Cluster 8 (Fig. 4.5).

Cluster 8 realises that there are no further clustering opportunities available because all records and clusters have achieved their preferred memberships, and the clustering process terminates.

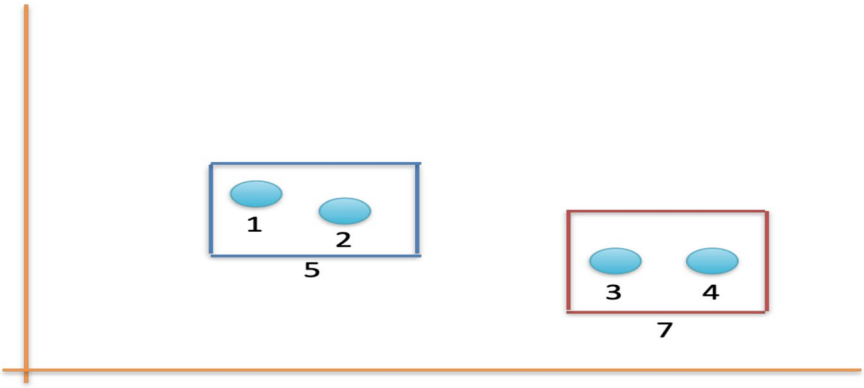


Fig. 4.4 Cluster 6 is destroyed; Cluster 7 is formed

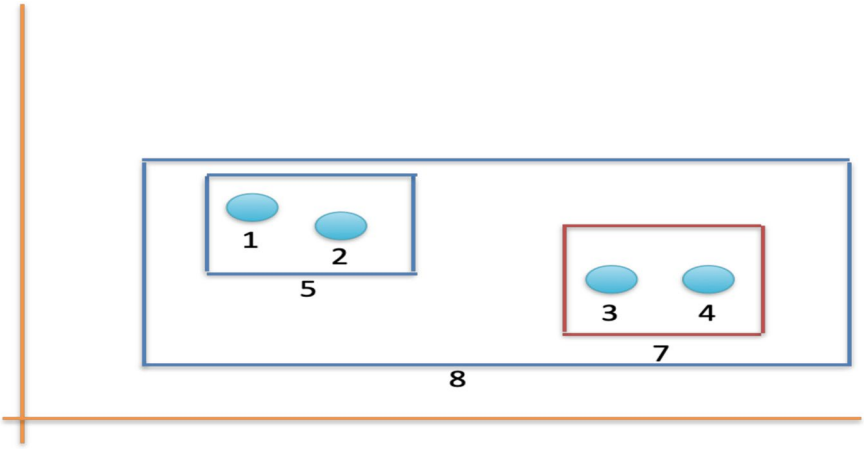


Fig. 4.5 Cluster 5 and 7 form Cluster 8

Microeconomics of Clustering

Agents representing records and clusters negotiate cluster memberships according to one of the several available models—the club model, the share model, and the tax model [3].

The ***club model***—data elements pay membership fees to join clusters. Fees are fixed.

The ***shareholder model***—data elements purchase shares in clusters. Share prices depend on cluster density and energy levels and may vary over time. Records can increase their energy levels by entering or leaving a cluster at

an opportune time. They can also lose energy if they make a poor clustering decision. This model increases differentiation among clusters in terms of their usefulness to users.

The ***tax model requires*** records to pay tax during their membership of clusters. This model enables evolutionary change in the system because records are forced to leave when they exhaust their energy levels, creating vacancies for newly arriving records. Differences in energy levels among data elements provide a mechanism for selection. This model encourages records and clusters to consider their long-term prospects when making clustering decisions. For example, a cluster that does not attract candidates for membership may reduce its membership tax to attract new members and thus prolong its life.

Data clustering and the resulting clusters depend on the chosen model of cluster membership. In particular, the following cluster features depend on the clustering criteria: cluster size—a large number of small clusters or a smaller number of large clusters; equality (all records are of equal importance) versus elitism (some records are given preference); and clustering speed.

An Example

Table 4.1 shows records available for clustering.

Suppose the club model is used, the cluster membership fee is set to three energy units (eu), and all records are assigned an equal amount of energy, say ten eu. In that case, the system will generate two clusters, as shown in Table 4.2.

Let us now consider the shareholder model. We shall assume that each record has an amount of eu equal to the purchase value, and that the amount of eu for participation in the cluster is calculated as follows:

Table 4.1 Records available for clustering

Buyer	Goods purchased	Value of the purchased goods
A	Car	500
B	Van	13
C	Car	10
D	Car	700
E	Truck	20

Table 4.2 Two clusters produced by the club model

Cluster name	Cluster members	Membership cost	Cluster energy level
CAR	A, C, D	3	9
VEHICLES	A, B, C, D, E	3	15

Table 4.3 Two clusters produced by the shareholder model

Cluster name	Cluster members	Membership cost	Cluster energy level
CAR	A, D	60	120
VEHICLES	B, C, E	2	6

The first record, A, can create a cluster called Car for 10% of the overall money (50 €).

When the fourth record, D, arrives in the system, they can decide whether to pay the same sum of 50 eu or a larger sum, say, 10% of their own overall money (70 eu). In the latter case, they will receive a larger number of cluster shares, which could later be sold when the cluster becomes richer and the record decides to leave the cluster and join another.

The average cost of entering the cluster is (50 + 70)/2 = 60 €.

It is not profitable for richer clusters to be shareholders in clusters with a low entry fee because those clusters are poor. The outcome is reflected in the data provided in Table 4.3.

We see a dramatic difference in how the two clusters are formed. The shareholder model separates the rich from the poor.

And what is the difference for the client?

The first model is useful for identifying which customers purchase which types of vehicles, and the second model can identify which customers purchase certain goods in large quantities. Together, the two models may help the client launch two distinct, focused advertising campaigns.

To summarise, the club model promotes equality and creates a larger number of clusters in earlier stages (because the membership fee is low and fixed, making it easier to create a new cluster); once in a cluster, records are reluctant to change their membership (because the energy level of new records is not sufficient to initiate the restructuring of clusters).

The shareholder model promotes rich records, generates fewer clusters, enables greater mobility of records even in later stages (because new rich records can force the restructuring of clusters, even ousting poorer records from rich clusters), and accelerates clustering (because the number of options available to each record is considerably reduced—the high entrance costs prevent poor records from joining rich clusters).

The tax model adds another dimension to the clustering process. When records must pay to remain in the system, the cluster structure evolves as impoverished records are forced to leave. The tax model is typically used alongside either of the other two models to drive the evolution of clusters.

Representing Clusters as Rules

It is often very convenient to represent clusters as rules of the type:

IF (condition A1) and (condition A2) and …. (condition An), THEN (condition B1) and (condition B2) and … (condition Bm).

An refers to conditions that include fields over which we have no control (independent fields), and Bm refers to conditions that encompass fields whose values we can manipulate.

An example of such a rule is: "If an order requires transporting a cargo weighing 5 kilograms, then this order should be allocated to the trucks of type G belonging to the ABC carrier company".

Note that a rule derived as described above always represents a cluster; therefore, if a clustering procedure discovered all clusters, it would have discovered all rules.

The inverse statement is not generally true—a cluster cannot always be represented by a rule, for example, when all records in the cluster belong to a single category (either all independent or all dependent).

Rules are evaluated using three criteria: representativeness, confidence level, and completeness.

Representativeness shows how many elements of a cluster are included in a rule. This parameter does not depend on the patterns.

Confidence Level shows how many elements of a cluster that are included in the left part of the rule (IF part) do not meet the conditions of the right part of the rule (THEN part). This parameter depends upon the pattern inherent in the cluster. For example, the pattern "among all ones that are parous, all are women" has a confidence level of 100%, while a reverse rule has a low confidence level because "not all women are parous".

Completeness shows how many elements of a cluster that meet conditions of the right part of the rule (THEN part) do not meet conditions of the left part of the rule (IF part). For example, the rule "if you are a human being, then you are mortal" has high representativeness but low completeness (since not all mortal beings are human beings).

The higher the value of representativeness and confidence level, the more valuable the revealed interdependency of cluster elements.

When a rule is derived, we can try to move conditions from the IF part of the rule to the THEN part of the rule. If this operation does not decrease the rule's confidence level, the modified rule is more useful. For example, the rule

"If an order requires transportation from Manchester, then destination location is London and the order should be assigned to truck of type ZA" is more useful than the rule "If an order requires transportation from Manchester to London, then this order should be assigned to truck of type ZA", provided that the confidence level has not been decreased by the operation.

Case Studies

Case studies in this section are summaries of real commercial systems designed and delivered by our team.

Adaptive Pattern Recognition for Transportation Scheduling

A logistics company required a tool capable of autonomously creating schedules that are as good as, or better than, those produced manually by an experienced operator. The customer did not define metrics to measure the proximity and similarity of schedules, so the evaluation of results was carried out by an expert. The customer provided data on 900 schedules.

The first task was to extract hidden rules from the customer-provided dataset. To this end, a table was created, with each row representing an order.

Our real-time pattern recognition tool identified 218 rules. Figure 4.6 shows the distribution of rules by confidence level.

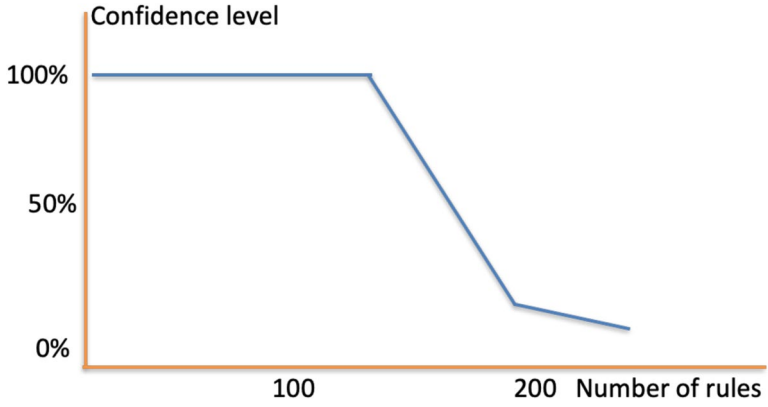


Fig. 4.6 Distribution of rules by confidence level

As the graph shows, more than half of the rules have a confidence level of 100%, indicating the tool's effectiveness.

The tool's rules were presented to an expert, who confirmed most of them and agreed that the identified dependencies are intrinsic to the problem domain. Moreover, the expert stated that 8% of the discovered rules were previously unknown to the system and that these rules had high confidence.

The discovered rules were then loaded into the knowledge base of our adaptive scheduler to compare a schedule produced by the scheduler without the extracted rules, a schedule produced by the scheduler with the extracted rules loaded, and a schedule produced manually by an experienced operator.

Test runs were executed on a dataset different from the one used to extract the rules, and each test run was based on a 1-week operational period. Loading the extracted rules increased the speed of automatic scheduling considerably: 3 h with rules versus 5 h without.

The schedule generated with the help of rules was more like the one produced manually by an experienced operator than the one generated without rules.

Loading the extracted rules into the scheduler improved the quality of the resulting schedule, as quantified below:

Manual rework needed decreased by 32%.
Journey quality increased by 17%.
The presence of gaps in the journeys decreased by 11%.
Fleet mileage decreased by 16%.
Fleet usage decreased by 8%.
The estimated time required for the customisation of schedules dramatically decreased from 1–2 months to 10–15 days.

Identifying possible consolidation options in transport logistics is one of the most useful applications of clustering and clustering analysis. For this purpose, appropriate filters are devised for the IF and THEN parts of the rules to pass only those fields relevant to the problem at hand; geographical coordinates, time windows, and ***Journey-Time Matrix*** information are typically of particular interest.

Each cluster passing through the filter can be considered a group of orders that are potentially amenable to consolidation.

At the next step, it is necessary to define, for each cluster, how consolidation should be applied—whether all orders in the group are shipped by a single truck or by several similar trucks. The decision depends on the distance between locations, since time is required to load and deliver each cargo, while

the driver's working hours are strictly limited. The decision also depends on cargo properties, as in some cases a special-purpose truck may be required to deliver chilled goods. Additional clustering runs on the consolidation clusters help surface further dependencies and details.

Further information on this adaptive road transportation scheduler is available in [4]. And tools for developing swarms of agents are described in [5].

Adaptive Pattern Recognition for Insurance

The Problem

Our client, one of the five largest insurance companies in the UK, faced the following problem. Because car insurance premiums depend on various parameters, including the client's gender and age, education level, yearly income, car class, and driving history, their lawyers have created over 25,000 documents related to car insurance contracts over the past 20 years.

The task given to our team was to analyse these documents, classify them by semantic similarity, and create a contract template for each group. Contract templates were expected to include the most frequently used clauses from the constituent documents within each group, to serve as a basis for all new contracts. Part of the task was also to analyse and classify available contracts from competitive insurance companies and to take these results into account when creating templates.

The initial estimate was that there would be around 100 groups of documents, and that the whole work would take approximately 16 man-years of highly qualified legal experts. Our task was to automate this process, thus saving time and money.

The Solution

The solution to this problem developed by our team [6] was based on the real-time clustering method described above, combined with the text processing method described in the next chapter.

First, our semantic processing system represented the document's meaning as a semantic network; then, our real-time clustering method was applied to classify documents; and finally, a heuristic method was developed to create templates based on groups of semantically similar documents.

Adaptive Pattern Recognition for Travel Forecasting

A travel agent had statistics on their customers that included personal information, travel history (previously visited country), and the country customers plan to visit next. The last item was often missing because not all customers articulated their plans when visiting the travel agent's website. The task was to help the travel agent forecast future destinations for their customers by discovering patterns in their travel datasets.

From available data, it was easy to derive rules that predict the behaviour of customers, such as:

IF Age range = "18–30", Previous Destination = "France" THEN the new destination = "Netherlands".

The pattern indicated that the company can expect customers from group 3293 to book their next visit to the Netherlands.

Adaptive Pattern Recognition for Fraud and Error Detection

Fraud detection is performed by identifying abnormalities or unusual activities in datasets, situations that deviate from general trends. Such discrepancies represent either intentional fraud or a data entry error.

For example, if most vehicles in a pool are charged from $10 to $25 for parking, while two trucks pay $200, there are at least two explanations for this discrepancy: a computer operator accidentally entered an additional zero, or a person in charge diverted some funds to his account and attempted to cover the fraud with false parking charges.

Discrepancies of this kind were identified for a client using our adaptive pattern recognition tool, which detects cases in which the value of an attribute in the THEN part of a rule deviates from the expected value.

Important Points

1. Adaptive pattern recognition by swarms of digital agents is unique because it is capable of discovering patterns dynamically, as data arrives, and is therefore highly suitable for discovering visitors' footprints at websites.

2. It represents a dramatic departure from conventional data mining methods, which require users to propose hypotheses that are then tested using mathematical methods.
3. A pattern recognition tool, based on a swarm of digital agents, can discover all hidden patterns in data autonomously through a process of agent negotiation.

References

1. G. Rzevski, P. Skobelev, Agent Method and Computer System for Negotiating in a Virtual Environment. Patent Reference No: WO 03/067432 A1. Published 14.08.2003.
2. Rzevski, G., P. Skobelev, I Minakov, Data Mining. Patent Reference No: GB 2411015 A. Published 17.08.2005.
3. Rzevski G., Skobelev, P., Minakov, I., Volman, S., "Dynamic Pattern Discovery using Multi-Agent Technology". *Proceedings of the 6th WSEAS International Conference on Telecommunications and Informatics (TELE_INFO '07),* Dallas, Texas, USA, March 22–24, 2007, ISBN: 978-960-8457-60-7, pp. 75–81.
4. Andreev, M., Rzevski, G., Skobelev, P., Shveykin, P., Tsarev, A., Tugashev, A. "Adaptive Planning for Supply Chain Networks". *Lecture Notes in Computer Science, Volume 4659, Holonic and Multi-Agent Systems for Manufacturing. Third International Conference on Industrial Applications of Holonic and Multi-Agent Systems, HoloMAS 2007*, Regensburg, Germany. Springer, ISBN 978-3-540-74478-8, pp. 215–225.
5. Rzevski, G., Skobelev, P., Andreev, V. "MagentaToolkit: A Set of Multi-Agent Tools for Developing Adaptive Real-Time Applications". *Lecture Notes in Computer Science, Volume 4659, Holonic and Multi-Agent Systems for Manufacturing. Third International Conference on Industrial Applications of Holonic and Multi-Agent Systems, HoloMAS 2007*, Regensburg, Germany, Springer, ISBN 978-3-540-74478-8, pp. 303–314.
6. Minakov, I., Rzevski, G., Skobelev, P., Volman, S., "Creating Contract Templates for Car Insurance Using Multi-Agent Based Text Understanding and Clustering". *Lecture Notes in Computer Science, Volume 4659, Holonic and Multi-Agent Systems for Manufacturing. Third International Conference on Industrial Applications of Holonic and Multi-Agent Systems, HoloMAS 2007*, Regensburg, Germany, September 2007, Springer, ISBN 978-3-540-74478-8, pp. 361–371.



5

Adaptive Semantic Processing

Semantic processing is adaptive if it can interpret unpredictably changing text.

Semantics is the study of meaning in communication. The word derives from the Greek semantikos, meaning "significant", from semaino, "to signify, to indicate", and from sema, "sign, mark, token". In linguistics, it is the study of how agents or communities interpret signs in particular circumstances and contexts.

With Swarm AI, we can enable computers to interpret text queries and provide meaningful written answers, in other words, to perform semantic processing.

Possible applications are numerous and include the following.

Digital translators
Text-based communication among computers
Semantic search engines
Semantic web
Semantic man-machine interfaces
Auto-abstracting, annotation, and classification systems
Document-flow management systems

Swarm AI performs semantic processing in a novel way: it treats it as the assignment of meaning to words.

An agent is assigned to each word in the text under consideration and to each possible meaning. Word agents and meaning agents negotiate until a consensus is reached on the meaning of each word.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

G. Rzevski, *Swarm AI*, https://doi.org/10.1007/978-3-032-26262-2_5

If a consensus cannot be reached, the conflict is resolved through an agent-triggered consultation with the user, followed by an update to the knowledge base.

Here is a simple example.

Let's assume the first word in a sentence is TABLE. The agent assigned to the word table looks it up in the dictionary and finds that it can mean an array of figures or furniture.

To resolve the issue, a second word is considered; it is CHAIR. The chair agent finds that the meaning could be a professor (looking at a table of figures) or a piece of furniture (a chair next to a table).

If the third word is LECTURE, the context becomes clear, and unique meanings can be assigned to words.

To simplify the process of extracting meanings, it is helpful to initially perform morphological and syntactic analysis of the text.

Definitions

A ***syntactic descriptor*** is a network of words linked by syntactic relations representing a grammatically correct sentence.

A ***semantic descriptor*** is a network of grammatically and semantically compatible words that represents a computer-readable interpretation of the meaning of a text. If semantic ontology describes all possible meanings of words in a domain, a semantic descriptor describes the meaning of a particular text.

Self-organisation, in the context of text understanding, is any autonomous change of a link between two agents representing different meanings of the same word.

Evolution, in the context of text understanding, is any autonomous update of the knowledge base with the newly acquired information.

The Meaning Extraction Process

The meaning extraction process is protected by patents [1, 2]. It consists of the following four steps.

1. Morphological analysis
2. Syntactic analysis

3. Semantic analysis
4. Pragmatics

The text is divided into sentences. Sentences are fed into the meaning extraction process one by one.

Morphological Analysis

An agent is assigned to each word in the sentence. Word agents access the knowledge base and acquire relevant morphological knowledge, enabling them to establish the characteristics of each word, such as noun, adjective, gender, number, case, time, etc.

If morphological analysis results in polysemy, i.e., a situation in which some words can play several roles in a sentence (noun, adjective, or verb), several agents are assigned to the same word, each representing one of its possible roles.

Syntactical Analysis

Word agents consult the knowledge base to acquire relevant knowledge of syntax, which enables them to identify the sentence's syntactic structure as follows.

A subject agent searches for a predicate agent of the same gender and number, and a predicate agent looks for suitable subject and object agents. Conflicts are resolved through a process of negotiation.

A grammatically correct sentence is represented by a syntactic descriptor.

If the results of the syntactical analysis are ambiguous, i.e., several variants of the syntactic structure of the sentence under consideration are feasible, each feasible variant is represented by a different syntactic descriptor.

Semantic Analysis

Each grammatically correct version of the sentence under consideration is subjected to semantic analysis.

Word agents access the knowledge base to learn possible meanings of the words they represent and, by consulting one another, attempt to eliminate inappropriate alternatives.

Once agents agree on a grammatically and semantically correct sentence, they create a semantic descriptor of the sentence.

If a solution that satisfies all agents cannot be found, agents compose a message to the user explaining the difficulties and suggesting how the issues could be resolved.

Each new grammatically and semantically correct sentence is checked for semantic compatibility with the semantic descriptors of preceding sentences. In the process, agents may decide to modify previously agreed semantic interpretations of words or sentences (self-organisation).

After all sentences are processed, the final semantic descriptor of the entire document is constructed, providing a computer-readable semantic interpretation of the text.

Pragmatics

Word agents access the knowledge base and acquire relevant knowledge on pragmatics, which is closely related to the application at hand.

At this stage, agents consider whether they need to execute any additional processes to satisfy the requirements of their application-oriented tasks.

Here are some examples.

If the application is a person-computer dialogue, agents may decide to ask the user for additional information.

If the application is a search engine, agents will compare the semantic descriptors of the search request with those of available search results.

If the application is a classifier, agents will compare the semantic descriptors of different documents and form groups of documents with semantic proximity.

Let us recapitulate the main features of the meaning extraction process.

Decision-making rules are specified in the knowledge base, which incorporates general knowledge on text understanding, language-oriented rules, and specific knowledge on the problem domain.

Every word in the text under consideration is allowed to autonomously and proactively search for its own meaning using the knowledge available in the knowledge base.

Tentative decisions are reached through consultation and negotiation among all parties.
The final decision on the meaning of every word is reached through a consensus among all the words.
Semantic descriptors are produced for individual sentences and for the whole text.
The extraction of meanings follows an autonomous trial-and-error pattern (self-organisation).
The process of meaning extraction can be regulated by modifying the knowledge base.

A Case Study of Adaptive Semantic Processing

During fierce competition among several research labs working to decode the human genome, my team received a request to develop a system capable of autonomously assessing whether any of the many daily-published abstracts were relevant.

A published abstract of a scientific paper, which needed to be converted into a computer-readable format using the meaning extraction process, is given below.

The title is "Cloning cassette containing the reporter gene xylE". The text reads as follows: "Two pUC-derived vectors containing the promoterless xylE gene (encoding catechol 2,3-dioxygenase) of Pseudomonas putida mt-2 were constructed. The t(o) transcriptional terminator of phage lambda was placed downstream from the stop codon of xylE. The new vectors, pXT1 and pXT2, contain xylE and the t(o) terminator within a cloning cassette, which can be excised with several endonucleases. When inserted into a transcribed sequence, this xylE cassette reports promoter activity and interrupts downstream transcription of the target sequence".

The semantic descriptor of the title of the abstract (Fig. 5.1) shows that the sentence has been completely understood by the system—the relations between the gene and locus and gene properties have been determined. In biology, a locus, by definition, is a specific site on a particular gene or chromosome; according to the domain knowledge base, "cloning cassette" is a synonym for the semantic concept "locus".

A semantic descriptor of a sentence is shown in Fig. 5.2. A tentative semantic descriptor of the whole text is modified during semantic analysis in a stepwise manner. As a result of the analysis of the last sentence, the system

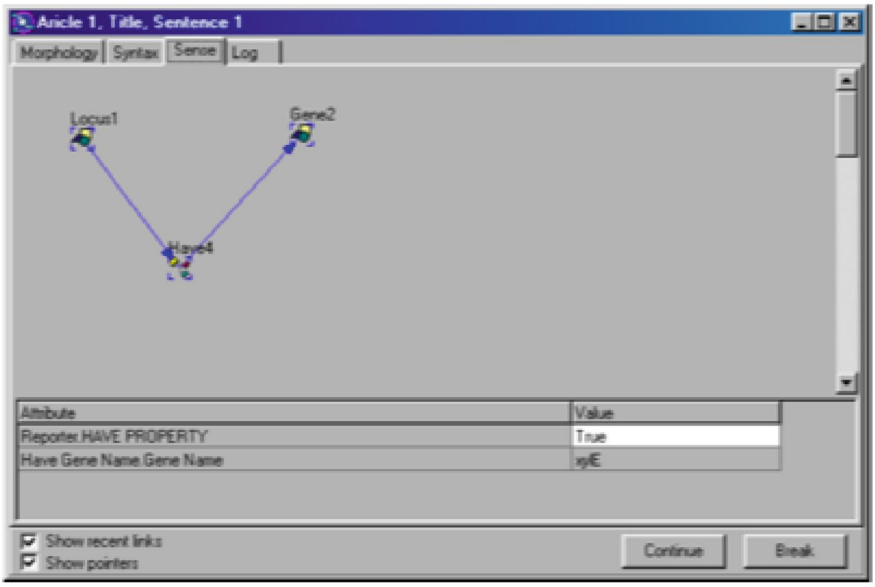


Fig. 5.1 A semantic descriptor of the title

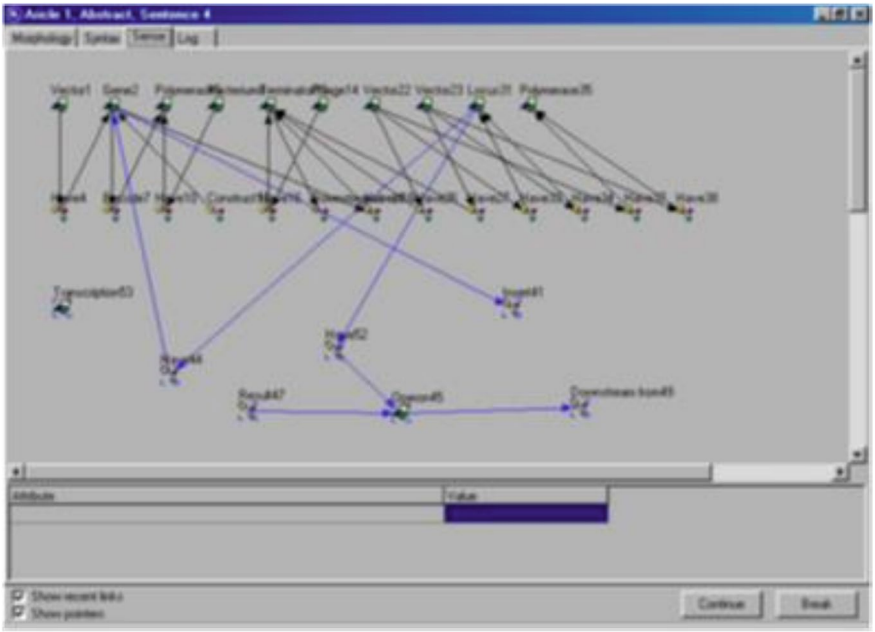


Fig. 5.2 Semantic descriptor of a sentence

discovered some new concepts and new relations between the existing nodes of the descriptor, including, a new relation "have" between the gene and the locus; furthermore, the gene has obtained a new insert relation and the relation "have" has been established between the locus and the new node, operon (in biology, operon is a controllable unit of transcription consisting of several structural genes transcribed together; it contains at least two distinct regions: the operator and the promoter; therefore, according to ontology and the text of the abstract, the semantic descriptor includes the concept "operon").

The meaning of the text is presented as a network of linked concepts. For example, the vector named "pUC-derived" has a gene with properties Promoterless = True, Name = xylE, Transcribed = True. The gene encodes the Polymerase named = 3-dioxygenase, which belongs to the bacterium = Pseudomonas putida. The gene also has a terminator with the property Transcribed = True, linked to the lambda phage, etc.

Comparison of the semantic descriptor with the original text discovered certain discrepancies, for example, the gene in the first sentence encodes the bacteria directly rather than via polymerase. Nevertheless, the semantic descriptor conveys the meaning of the abstract clearly enough.

Now, let us look at pragmatics. In addition to creating semantic descriptors for each abstract, it is necessary to formulate a descriptor of the enquiry. All the coefficients here are relative rather than absolute, which means that it is impossible to determine if the rating of 100 is good or bad—it all depends on the particular request. All the comparisons are made based on rules specified in ontology. A change in ontology may change the ranking of semantic descriptors.

Semantic Web

Tim Berners-Lee originally expressed his vision of the semantic web as follows:

> I have a dream for the Web [in which computers] become capable of analysing all the data on the Web—the content, links, and transactions between people and computers. A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy, and our daily lives will be handled by machines talking to machines. The intelligent agents, people have touted for ages, will finally materialise (Tim Berners-Lee 1999).

The Internet is a vast digital network of computers spreading around the globe. There is no exact real-time count of everyone on the Internet, but as of October 2025, approximately 6.04 billion people were online, representing about 73.2% of the world's population. This figure is constantly changing, but the number of Internet users continues to grow globally. One could argue that nearly three-quarters of the people on Earth live in a global village—they can communicate rapidly with each other, exchange gossip, share photos, trade, provide services, and ask for help.

The real breakthrough in establishing a useful global network came with Tim Berners-Lee's invention of the World Wide Web in 1989. The web is a network of documents in a standard format, stored on interconnected computers. Its significance lies in its ability to link documents directly, regardless of their location.

As of 2025, the indexed web is estimated to contain at least four to five billion pages, while the total number of documents and files across the entire Internet likely numbers in the trillions.

The next stage is to build a network of web content—the semantic web—that enables machines to interpret the meaning of data and to satisfy requests from people and machines to use that content.

The intelligent agents of Swarm AI are those referred to by Tim Berners-Lee as capable of performing semantic analysis of web document content [3]. The swarms of agents can interpret the content of documents written in a natural language such as English.

English to Sinhala Machine Translation

During my visits to Sri Lanka as a Visiting Professor at Moratuwa University, I have contributed to developing the first-ever English-to-Sinhala dictionary using Swarm AI technology [4, 5]. The dictionary was developed using morphological, syntactic, semantic, and pragmatic analysis, as described above.

Important Points

1. Text interpretation by swarms of digital agents is unique because it performs semantic processing step by step—word by word and sentence by sentence—allowing the text to be unpredictably altered during processing.
2. A swarm of digital agents can read scientific abstracts instead of researchers and select for them the relevant texts.

3. The Swarm AI will be instrumental in building the semantic web.
4. Adaptive semantic processing is a very effective method for language translation.

References

1. G. Rzevski, P. Skobelev, Agent Method and Computer System for Negotiating in a Virtual Environment. Patent Reference No: WO 03/067432 A1. Published 14.08.2003.
2. Minakov, I., Rzevski, G., Skobelev, P.: Automated Text Analysis. Patent Application No. 305634, UK (2004).
3. Rzevski, G., Skobelev, P., "Agent Based Semantic Web", Semantic Universe Website, 2009.
4. Hettige, B., Karunananda, A. S., Rzevski, G. "MaSMT: A Multi-Agent System Development Framework for English-Sinhala Machine Translation". International Journal of Computational Linguistics and Natural Language Processing, Volume 2, Issue 7, July 2013, ISSN: 2279 – 0756.
5. Hettige, B., A. Karunananda, G. Rzevski, "Thinking like Humans: A New Approach to Machine Translation". International Conference of the Sri Lanka Association for Artificial Intelligence, 2018, pp. 256–268. Published by Springer Singapore.



6

Adaptive Engineering Products

There are many ways in which swarms of agents can improve the design and performance of engineering products by making them adaptive.

We shall consider four cases.

1. Replacing rigid control systems with swarms of digital agents.
2. Swarms of smaller engineering products as a more cost-effective option than large monolithic products.
3. Using swarms of digital agents to manage the lifecycle of large aerospace products.
4. Helping with the design of Airbus A380 wings.

Axial Turbo Compressors with Adaptive Geometry

The idea of adaptive control by a swarm of agents will be demonstrated using the work of one of my best PhD students, Gwyn Morgan [1].

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

G. Rzevski, *Swarm AI*, https://doi.org/10.1007/978-3-032-26262-2_6

The Problem

Axial turbo compressors are used in jet engines, industrial gas turbines, gas pipeline pumping, and in process plants. Their performance is limited by stall and surge, which make the gas or air flow unstable and even reverse direction, resulting in significant damage.

The Russian supersonic aircraft TU 144 was lost at the Paris Air Show in 1973 due to a surge during extreme manoeuvres.

Current Practice

Current practice is to limit the scope of compressor operation to provide large stall and surge safety margins.

The Solution

An axial turbo compressor was designed with variable geometry and a digital agent controlling each movable part. Agents were organised into a swarm and empowered to negotiate with one another the relative positions of stator vanes, with a view to achieving a performance as close as possible to optimal under continually changing aerodynamic conditions.

Sensors were introduced to monitor the aerodynamic conditions in the compressor, and particularly around each movable element. Agents picked up signals from sensors and, after consulting each other, determined the best vane positions under ever-changing aerodynamic conditions. Then actuators moved the vanes accordingly. The swarm consisted of six vane agents and an efficiency agent, whose role was to negotiate with vane agents the corrections which would improve compressor efficiency without unduly affecting the flow, as shown in Fig. 6.1.

The implications of turning a mechanical compressor into a mechatronic system, with sensors, actuators, digital hardware, software, and AI, were extraordinary. It became feasible to achieve critically important functions.

Self-diagnosing by monitoring compressor conditions and identifying faults.
Self-repair by isolating failed parts and making them harmless.
Graceful degradation of performance by repositioning remaining healthy parts to achieve a reduced but acceptable performance.

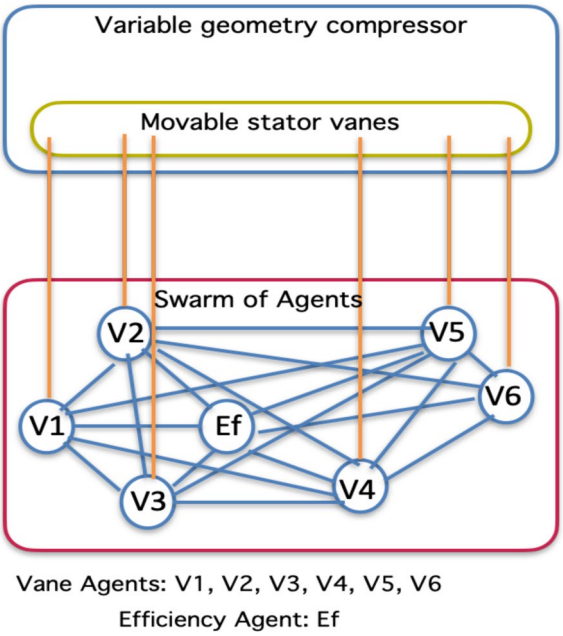


Fig. 6.1 Swarm of agents controlling movable parts of a compressor

Simulation

Simulation is an essential part of engineering design, enabling designers to evaluate the performance of the proposed product before significant resources are spent on building a prototype.

Simulation (see Fig. 6.2) confirmed the effectiveness of adaptive compressor control in achieving two main objectives.

Maximising the compressor's operating range.
Maximising the efficiency of the compressor.

Both objectives were achieved primarily through the swarm of agents' ability to rapidly react to changes in operating conditions and adjust vane angles accordingly.

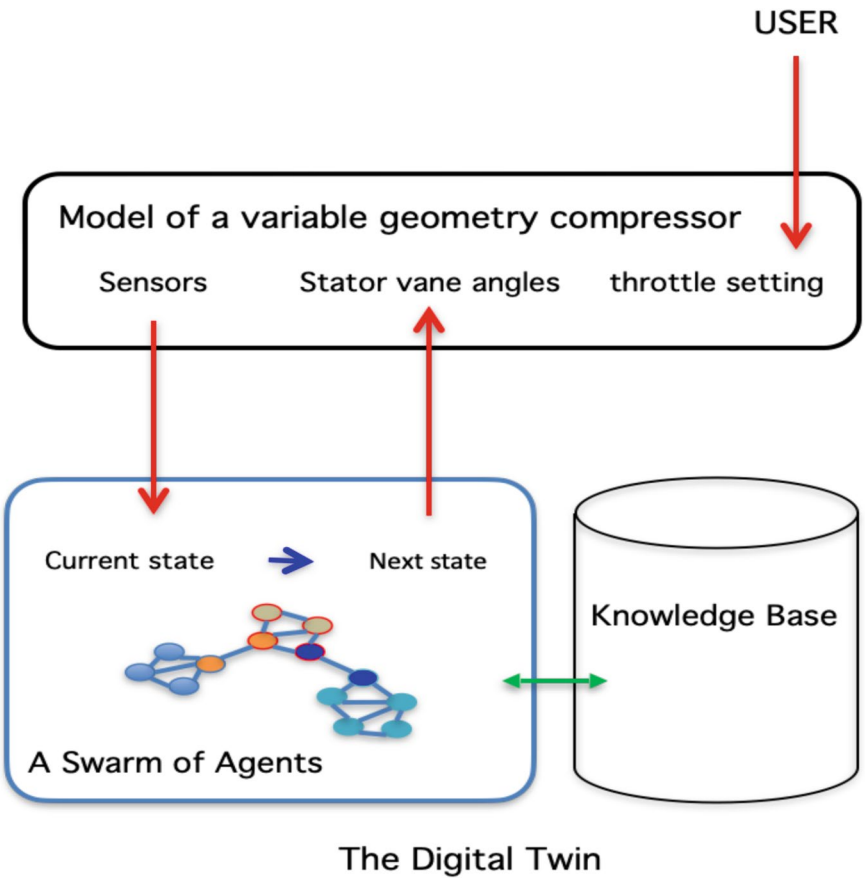


Fig. 6.2 A simulator for the evaluation of the adaptive control of stator vanes

A Swarm of Smaller Space Robots Versus a Single Large Robot

A UK team launched a robot to explore Mars. Unfortunately, it got lost. It disappeared without a trace amid speculation that it landed in a crevice on the Mars surface and lost communication with the control station.

An American robot landed safely on the surface of Mars but could not survive more than a couple of weeks because space dust covered its solar cells.

Both disasters could have been avoided by sending a family of smaller robots to Mars.

In fact, just before the unfortunate Mars robot project collapsed, I joined a team of researchers from three universities and two UK spacecraft manufacturers, focused on developing technologies for autonomy and robustness in space.

Following the design principles described in [2], our team concluded that a family of five much smaller intelligent robots would perform better under uncertain conditions in space than a single large robot.

Each member of our robot family had limited intelligence and was capable of undertaking simple tasks, such as placing scientific instruments in the correct location and providing a variety of services to other members of the family, e.g., cleaning their solar cells if they became covered by space dust and helping them to get out of cracks in the Martian surface.

A control system, based on a swarm of digital agents, exhibiting collective intelligence greater than the sum of the intelligence of individual robots, had the capacity to adapt to an unknown Mars environment.

The cost and weight per unit performance for the family were estimated to be lower than those of an equivalent single robot. In addition, five smaller robots offered an important advantage in packaging for launch and delivery.

The results of this research were ignored with catastrophic consequences.

Adaptive Lifecycle Management of Large Aerospace Products

The lifecycle of a large aerospace product, such as an airliner or a military aircraft, consists of a very large number of interconnected activities, which are normally partitioned into several phases.

Design
Production
Operation
Decommissioning

Each lifecycle phase consists of activities as outlined below.

Design consists of all activities needed to produce, operate, and recycle a product, including: market research, requirements specification, conception, design management, overall design (including design for X, where X is performance, safety, reliability, operability, manufacturing, assembly, maintainability, and recycling), and design of components.

Production encompasses the selection and development of production technologies, production planning and management, supply of materials and components, manufacturing, assembly, testing, and quality assurance.

The operation includes the scheduling and management of aircraft and their crews within a fleet; maintenance and repair tasks and resources; and logistics support tasks and resources.
Decommissioning includes the safe disposal and recycling of components and/or materials.

The aircraft industry's safety record is excellent. This research aimed to identify areas where lifecycle costs could be significantly reduced by improving the management of lifecycle complexity while maintaining or improving current safety levels.

The Problem

A comprehensive failure analysis [3, 4] has established several surprising features.

Each aircraft has a unique failure pattern, which depends on how it is used, how it is maintained, and how it is supported. Operational conditions play a significant role: aircraft that are often scheduled to fly to the Middle East will be subject to failures caused by sand and high ambient temperatures, while aircraft regularly scheduled to fly to the North will suffer from different types of failures due to low temperatures, snow, and ice.
Aircraft failures are interdependent; a failure in one component may cause failures in others, depending primarily on proximity and functional links. A typical example is placing all operational and redundant components of a hydraulic system together, which can result in all of them failing as a result of a single external event.
Timings of component failures are unpredictable but not random; patterns of failures can be discovered from failure data; these patterns are not permanent—they change as operating conditions of aircraft change and with the age of aircraft.

Let us illustrate the complexity of the aircraft lifecycle, taking as an example the B747 airliner, one of the largest aircraft in the world, which began operations in 1970 and has been sold to 170 airline companies. A B747 has six million components, of which three million are rivets. Aircraft have 5000–6000 safety, reliability, and maintenance-critical modules, which

need to be regularly checked, tested, calibrated, cleaned, repaired, or replaced. To safely deliver one flying hour, it requires 12 maintenance man-hours (compared with Concord, which required 137 man-hours for each hour of flying).

Current Practice

The current practice is to consider each phase of the lifecycle separately, managed by different teams of specialists, often from different organisations, with limited interaction between them. As a rule, there is no central lifecycle management team responsible for coordinating all activities, from the conception of the future product to its disposal and recycling. Moreover, each lifecycle phase is normally supported by standalone computer systems, such as computer-aided design (CAD), computer-aided engineering (CAE), computer-aided manufacturing (CAM), enterprise resource planning (ERP), and various production and logistics schedulers, which are rarely fully compatible with each other and never integrated into a single lifecycle system.

Partitioning lifecycle activities into separate phases and managing each phase with distinct management systems, supported by standalone computer systems, weakens important links between key lifecycle activities, thereby delaying the flow of information, particularly feedback from manufacturing, assembly, operation, maintenance, and support to design. As a consequence, designers have no real-time access to data on failure analysis, which would help them better understand the relationship between failures and design decisions.

Further, the current practice is to assess reliability and organise identical maintenance for all aircraft manufactured to the same design, despite each aircraft having a unique life.

Failure rates are calculated ignoring spatial interdependence.

Maintenance procedures are prescribed for the whole life of aircraft, ignoring the fact that the patterns of failures change over time.

The Solution

A solution to the lifecycle management problem, developed by our team, is based on the assertion that a complex system can be managed only by a complex system [3, 4].

The new adaptive lifecycle management (ALM) system has been designed as an adaptive network of swarms of digital agents, rather than the currently used monolithic computer programs. Nodes of this network are lifecycle services, which are developed independently, one by one, and then configured into the adaptive lifecycle management system.

The requisite granularity for an ALM service is the ability to identify, store, and process every single constituent component (millions of them), as well as all design, production, and operational resources.

The service of this size cannot possibly be designed as a single integrated unit. The trick is to discover processes that are reasonably self-contained and form elementary services and to design them as swarms of agents, ensuring that these services cooperate and/or compete through negotiation.

The requisite complexity for an ALM service is the ability to adapt to every change of state in the product lifecycle.

This requirement contradicts the traditional practice of simplifying complex systems by partitioning them into self-contained constituent subsystems, which are then considered separately, one by one, an approach known as reductionism.

Reductionism is, of course, valid for deterministic systems and may work for systems of very low complexity, but for levels of complexity that the aerospace business currently experiences, the approach is not valid. Instead of simplifying reality, we must build complexity into management systems to match the complexity of their physical reality and the corresponding economic environment.

Complexity has been built into the ALM service by giving constituent agents partial autonomy, mandating that they negotiate how to solve each task, requiring agents to consult the lifecycle knowledge base before undertaking any action, and ensuring that links among agents can be easily modified when required. Methods used for designing complexity into ALM service are described in detail in [5, 6].

An ALM service of sufficient granularity and complexity can be implemented through an ontology-driven swarm of digital agents, as shown in Fig. 6.3.

The lifecycle knowledge base contains all conceptual knowledge stored in ontology and the data required to manage the whole lifecycle.

The digital twin of the aircraft lifecycle comprises a network of agent swarms, each swarm dedicated to a specific lifecycle management function (strategic function, design, production, operation, and decommissioning).

Each swarm is designed to communicate with the others and to access the lifecycle knowledge base via the lifecycle network.

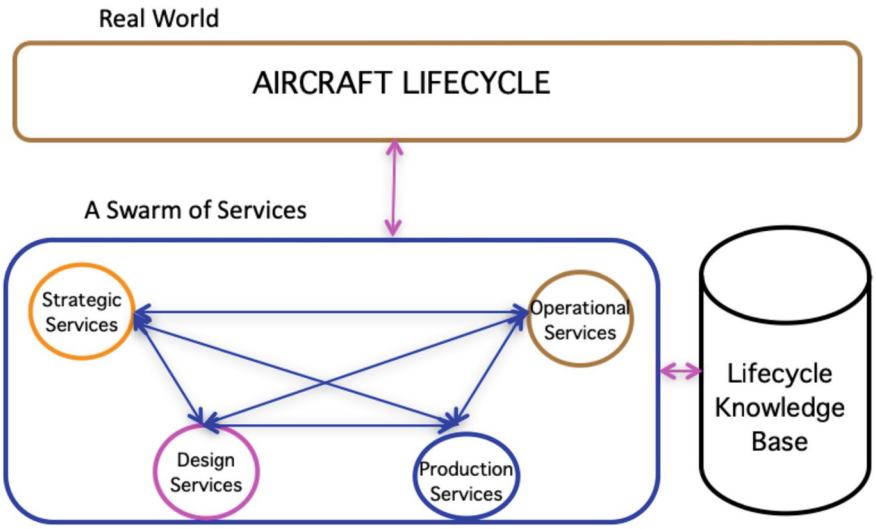


Fig. 6.3 Architecture of an adaptive aircraft lifecycle management system

Swarms can be located at different geographical locations.

There is a fundamental difference between the design phase of a product lifecycle and all the other phases (production, operation, and decommissioning). Design is about creating a product and its in-service life that do not yet exist. At the beginning of the design phase, we have no components, configuration, or ontology. What we have is an incomplete set of candidate product elements (components, configurations, and ontology), which may be used during the design phase.

Design activities can then be divided into two categories.

Creating new product elements.
Selecting existing product elements.

While the creation of new elements is currently best done by human designers, the selection of existing aircraft elements can benefit from AI. We should not overlook the importance of design evaluation. Here, we can again distinguish two categories:

Evaluating how design meets lifecycle specification and aesthetic criteria.
Evaluation of design consistency (including checking for possible clashes when a large number of different designers do the design).

And again, the first category requires input from human design evaluators, while the second category is more efficiently performed by AI software.

The development of an ALM system is a gradual, evolutionary process. The initial knowledge base can be rudimentary and expanded as required. Elementary swarms can be added one by one, enabling the system to be used and tested in real-life applications from the very beginning of its development and to grow as required. The following ALM services have been specified to form the main nodes of the ALM system:

Adaptive strategy services
Adaptive forecasting service
Adaptive long-term planning service
Adaptive design service
Adaptive component selection service
Adaptive design consistency service
Adaptive design knowledge discovery service
Adaptive production management service
Adaptive supply chain scheduling service
Adaptive production scheduling service
Adaptive operation management service
Adaptive failure forecasting service
Adaptive aircraft maintenance and repair scheduling service
Adaptive personnel maintenance and repair scheduling service
Adaptive aircraft support and resource scheduling service
Adaptive spare parts storage and scheduling service

This list is not exhaustive. It includes services that are most likely to improve productivity and reduce the costs of aerospace product lifecycles. The decommissioning phase will be considered at a later stage.

The above ALM Services are, in turn, composed of elementary services listed below.

Adaptive human resources scheduling service
Adaptive physical resources scheduling service
Adaptive transportation scheduling service
Adaptive loading service
Adaptive routing service
Adaptive knowledge discovery service

Adaptive simulation service
Adaptive components selection service
Adaptive design consistency service

Each constituent service of the proposed ALM system must be complex adaptive, i.e., it must be able to self-organise in reaction to any disruptive event, aiming to eliminate or, at least, reduce consequences of the disruption before the next disruptive event occurs. Since the task of services is to allocate resources to demand, and disruptive events are typically related to unexpected changes in demand or resource availability, self-organisation will always result in rescheduling.

Helping with the Design of Airbus A380 Wings

The Problem

The A380 wing comprises approximately 10,000 parts, 2000 assemblies, and 300,000 relations between parts; therefore, its design must be undertaken by a large team of designers. At the boundaries between sections designed by different designers, mismatches, known as "clashes", occasionally occur. Clashes are notoriously difficult to detect, and undetected clashes cause severe problems in later stages of product development.

Current Practice

Experienced designers look for clashes manually. Alternatively, a network of large mainframe computers is employed to detect clashes by exhaustive search, which could last 3 weeks.

The Solution

A swarm of agents performing clustering and ontology-based knowledge of how to detect clashes was designed and prototyped.

Results

Time to find clashes has been reduced from weeks to 3–6 h. Design problem areas have been identified.

Important Points

1. The performance of an engineering product that works in a complex environment can be considerably improved by replacing traditional rigid controls with complex adaptive control systems, such as a swarm of digital agents.
2. A swarm of smaller engineering products, controlled by a swarm of digital agents, is more effective than a single large product.
3. Current lifecycle management of large engineering products is out of date. A considerable savings would be achieved by using Swarm AI to manage its complexity.

References

1. Morgan, G., Rzevski, G., Wiese, P.: "Multi-Agent Control of Variable Geometry Axial Turbo Compressors". Journal of Systems and Control Engineering, issue I3 vol. 218 (2004), ISSN 0959-6518, pp. 157–171.
2. Rzevski, G., P. Skobelev, "Managing Complexity". WIT Press, Southampton, Boston, 2014. ISBN 978-1-84564-936-4.
3. Rzevski, G., J. Knezevic, P. Skobelev, N. Borgest, O. Lakhin, "Managing Aircraft Lifecycle Complexity". International Journal of Design & Nature and Ecodynamics. Volume 11, No 2 (2016), pp. 77–87. ISSN: 1755-7437.
4. Rzevski, G., "Managing Machine Functionability Using Methods of Complexity Science", 2013 Annals of Mirce Mechanics, pp. 68–76, Mirce Akademy, Exeter, UK, January 2014.
5. Knezevic, J., "Reliability, Maintainability and Supportability: A probabilistic Approach". McGraw-Hill, 1993
6. Knezevic, J., "Systems Maintainability: Analysis, Engineering and Management," Springer, 1997.

7

Adaptive Enterprise

Introduction

An enterprise is adaptive if it is capable of rapidly detecting unpredictable disruptive events, identifying affected parts of the enterprise, and self-organising to eliminate, or at least minimise, the negative consequences of the disruption without disturbing the unaffected parts of the enterprise.

Enterprise adaptability can be achieved by building its digital twin and empowering it to allocate resources to demand [1].

The Internet-based global market has reached such a level of complexity that only adaptive enterprises can expect to prosper.

Let's consider an example of the adaptive enterprise.

Adaptive Manufacturing

Cars are currently manufactured using rigid production lines, which are very expensive to design and build and have a short lifespan.

The author and his research team have developed technology to replace rigid, expensive car production lines with self-organising, evolving car production ecosystems, which are particularly suitable for the production of electric cars.

The production infrastructure comprises several key elements.

Assembly Points, where components are assembled into subassemblies and subassemblies are assembled into cars.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

G. Rzevski, *Swarm AI*, https://doi.org/10.1007/978-3-032-26262-2_7

Assembly Robots, which assemble cars in cooperation with assembly workers.
Stores, where components and subassemblies are stored while waiting to be assembled.
Mobile Robots, which load, unload, and transport components and subassemblies between stores and assembly points.

Assembled electric cars are driven from assembly points to Pre-Delivery Stores.

Every car component and every subassembly is equipped with an electronic tag with sufficient processing capacity to maintain a component/subassembly agent and local ontology.

Digital agents communicate with each other via the digital network, which is connected to the Internet.

Assembly Points have an extended digital processing facility for handling, storing, modifying, and displaying car designs.

The production progresses by agents sending messages to each other, triggering the delivery of components/subassemblies to assembly points.

Such a production ecosystem has the following advantages over current mass-production lines.

The proposed system is a socio-technological system (rather than purely technological or social); constituent agents have either human intelligence (HI) or artificial intelligence (AI).

The decision-making is distributed (rather than centralised); constituent agents are empowered to make certain decisions and may cooperate or compete with each other.

The manufacturing process is adaptive (rather than rigid, hierarchical, or command-and-control), capable of autonomously eliminating, or at least reducing, the consequences of unpredictable disruptive events.

The manufacturing process is self-organising (rather than organised and, from time to time, reorganised by managers).

The manufacturing is sustainable, capable of coevolving with its environment over a long period of time.

The manufacturing is scalable, capable of growing in size and/or performance.

Versatile Manufacturing

To bring manufacturing closer to points of demand for manufactured goods, it is necessary to design smaller manufacturing plants in which Swarm AI takes over all operational decision-making from HI.

The cost-effectiveness can be further increased by ensuring that the plant manufactures a range of similar products.

For such ***versatile manufacturing*** to be possible, we must have a wider range of movable resources and a swarm of agents attached to ontology with classes of objects from domains of the full range of products to be manufactured [2].

When orders arrive for a product that is not currently in production, swarms of agents will rapidly assemble a production line for the required product.

Virtual Enterprise

We shall consider here a more radical approach to adaptability, usually called a virtual enterprise (VE).

A virtual enterprise is a network of interconnected autonomous units, each unit being an organisation (such as a design bureau or a manufacturer) or an individual (a freelance programmer or an architect) [3].

Links between the units may be autonomously and dynamically established, dropped, and re-established depending on the VE needs (self-organisation).

There is considerable uncertainty about the capacities required for non-core activities; therefore, each VE tends to have a partnering arrangement with a larger number of companies or individuals than necessary (redundancy), just in case some of them are busy when their services are needed.

Since all semi-permanent partners operate in the same highly volatile environment, it is in their own interest to be interconnected with many potential purchasers of their services. As competitive conditions change over time, VEs may permanently drop connections with some of the old constituent units and seek to establish links with new units (evolution).

It is assumed that VEs will coexist and either compete or collaborate with other VEs. They would all try to improve their performances concurrently in relation to one another (coevolution), creating an ever-changing environment.

Not every VE will get its game right—there are always companies that underperform and those that disappear. New companies are formed, enter the market, and compete for the requisite number of connections.

Virtual enterprises, when considered at the level of individual resources, including elements of organisational knowledge, working capital, human resources, machines, work in progress, computers, and buildings, behave as

complex adaptive systems, and they are therefore eminently suitable to operate in complex markets, such as the Internet-based global market.

The key issue in designing a VE is then to determine the appropriate level of uncertainty, which must be built into its organisational structure to keep the system far from equilibrium and prevent it from relapsing into equilibrium or descending into chaos.

Agent-Based Model of a Virtual Enterprise

Our team has developed a framework for modelling VEs that fundamentally differs from traditional mathematical and probabilistic approaches. The new model is ontological, that is, it depicts a VE in terms of classes of constituent objects and their attributes, relationships, and scripts.

The ontological approach produces models that closely mirror the real organisation and preserve its inherent variety and uncertainty. It is of considerable advantage that ontology of an adaptable organisation can be built by domain experts without any knowledge of advanced mathematics or computer programming. Once the ontology is constructed, the model can be used to evaluate the VE's ability to meet a given level of demand through its constituent resources.

There is a choice of levels of detail that can be incorporated into the model. Digital agents can be assigned to groups of demands and resources (e.g., a project or a plant) or to each demand, such as an order or enquiry, as well as each organisational resource (e.g., an employee, a manager, a machine, a unit of capital, or an element of organisational knowledge). In any case, the allocation of resources to demands is achieved through negotiation between agents representing demands and resources, rather than through centralised algorithms.

The agent-based modelling approach described here is, to the best of our knowledge, novel. It is highly flexible and can be used for planning VE operations and allocating its resources to orders.

Agent-Based Simulator of a Virtual Enterprise

To simulate the operation of a networked VE, we assign an agent to each resource within the organisation, to all partner organisations connected to the organisational network, and to every enquiry or order arriving from customers.

Through negotiation among agents representing the demands (enquiries/orders) and those representing the resources, a deployment schedule is derived for each order.

The software is designed to accommodate hundreds of thousands of agents working in parallel and is therefore capable of simulating sizable networks and exhibiting emergent properties. The agent negotiation process is driven by a generic swarm engine that supports a wide variety of simulators and schedulers.

Given a known order distribution, the system can be used to determine the types and quantities of resources required to meet projected demand. It can thus be used to determine the number of constituent units of a VE that provide a sensible level of redundancy to protect against low-capacity risk. The dynamics of connecting, disconnecting, and re-connecting constituent units (self-organisation) can be simulated, as can the process of abandoning some old units and linking with new resources (evolution).

Agent technology provides possibilities not available in conventional simulators. The swarm of agents, which is a constituent component of our VE model (the other component being a knowledge base), provides a means of allocating VE resources (human skills, servers, and methodologies) to demands (orders, projects, tasks, and enquiries).

One workstation can handle up to 400,000 resources/demands. If a larger number is required, a network of workstations can increase capacity by an order of magnitude (i.e., by a factor of 100). Agents can simulate the autonomous behaviour of each networked unit, including co-operation strategies, redundancy policies (risk management), special offers, and capacity limitations. Above all, an agent-based simulator represents a virtual society of proactive, intelligent participants that reflects real-life VEs.

In highly dynamic business situations, simulators based on distributed decision-making mechanisms, such as our agent-based simulator, offer a distinct advantage by enabling rapid renegotiation of previously agreed decisions and thus accommodating unpredictable changes in demand and supply.

The effectiveness of an agent-based simulator is to a considerable extent determined by agent negotiation strategies. These strategies are, in turn, specified by negotiation protocols, which define rules for constructing and dispatching messages and are stored in ontology.

Let us consider an example of protocols for negotiation between a project agent (demand agent), PA, and a programmer agent (resource agent), RA. The agent PA begins the negotiation process by enquiring about the programmer's skills. If the answer is satisfactory, the agent PA enquires about the programmer's experience. Assuming that the answer is again satisfactory, negotiations turn to the deadlines and available capacities. The agent RA may find it

necessary to interrupt negotiations with PA to secure a delay in delivering another, less important task. Once that negotiation is complete, the agent RA returns to further discussions with PA.

Our simulator incorporates specialised agents into its swarms that manage the skills of VE employees. Their task is to monitor and improve their clients' employability by recording rejections due to inadequate skills or experience when applying to participate in projects and by proposing and scheduling appropriate training. Skills agents can negotiate training places for clients with internal or external training providers and recommend distance-learning courses.

To increase the flexibility of agent-based VE simulators, an editor has been developed that allows domain knowledge and negotiation protocols to be modified via company websites.

Simulator users without programming knowledge can adjust ontology elements to suit their specific requirements without involving simulator designers.

Important Points

1. The complexity of the Internet-based global market is increasing. The consequence is the unpredictability of supply and demand. Adaptive enterprises have the best opportunity to prosper under conditions of uncertainty—they can rapidly adapt to any unpredictable change.
2. To bring manufacturing back to the West, close to the points of demand for manufactured goods, we must design versatile manufacturing plants in which a swarm of agents rapidly switch from producing one type of product to another, within the range.
3. Simulation of an unconventional adaptive enterprise—a network of semi-permanent links among many organisations and individuals, managed by swarms of digital agents—demonstrates its suitability for operating in conditions of high uncertainty.

References

1. Rzevski, G., "The Future is Digital: How will Complexity and AI change our Lives and Work". Springer Nature, 2024.
2. Rzevski, G. "Harnessing the Power of Self-Organisation" International Journal of Design & Nature and Ecodynamics, Volume 11 No 4 (2016), pp. 483–494. ISSN: 1755-7437.
3. Rzevski, George, Skobelev, Peter, Korablin, Mikhail: Multi-Agent Models of Networked Organisations. Proc. of Intern. Workshop "New Models of Business: Managerial Aspects and Enabling Technology", School of Management of Saint Petersburg State University, Russia, June 28–29, 2001, pp. 46–49.

8

Adaptive Society

A society is adaptive if it can detect unpredictable, disruptive events arising from its geopolitical, socioeconomic, or military environments, identify affected parts, assess potential damage, and self-organise to eliminate or at least minimise the negative consequences of disruption (e.g., hacking or military attacks) without disturbing parts that were not affected.

In an adaptive society, the key economic resource is not capital; it is knowledge and intellect, human and artificial [1].

The most productive jobs are those in which knowledge workers create, process, and exchange knowledge.

Wealth is produced by trading in knowledge rather than goods, and the created wealth is sufficient to cost-effectively feed, water, house, educate, maintain in good health, move, and employ its members without endangering its environment.

Only a complex society can be adaptive.

Complexity and Social Systems

For these discussions, let us agree that any system in which dominant agents are people (a family, a club, an association, and a country) is a social system. Social systems include socioeconomic systems (a business and a market); socio-technical systems (transport and a supply chain); socio-political systems (a nation and a union of nations); and urban systems (a town and a metropolis), to mention just a few.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

G. Rzevski, *Swarm AI*, https://doi.org/10.1007/978-3-032-26262-2_8

Let us apply the seven criteria of complexity to social systems to test if they are complex.

Connectivity

Members, social agents within social systems, are interconnected, interdependent, and engaged in rich interaction.

Autonomy

The degree of autonomy of members in a social system is the most important criterion for distinguishing between complex/non-complex systems.

If members are always instructed what to do, we have a rigidly structured, rather than a complex, social system (a command-and-control hierarchy, a dictatorship).
If members have complete freedom to behave however they want, we have social chaos (a spontaneous riot, anarchy).
If agent autonomy is substantial but not total, the social system is complex.

In reality, the autonomy of agents in social systems is never complete; it is always constrained by law, social norms, and conventions and by a bewildering array of rules, regulations, and policies.

It follows that social systems, with a few exceptions, are complex systems.

Emergence

In general, the global behaviour of social systems emerges from the interactions of members and is therefore unpredictable, though not random. The uncertainty of the global behaviour of social systems is directly proportional to the autonomy of constituent agents—in rigidly structured social systems (strong dictatorships), emergence may be close to zero.

Nonequilibrium

In a complex social system, disruptive events are always present, and their frequency depends on the complexity of the environment in which the system

is embedded. As we entered the highly interconnected and interdependent Internet-based global society, the so-called global village, the frequency of disruptive events (unpredictable changes in family membership, associations, schooling, employment, earnings, leisure pursuits, housing costs, etc.) increased to such a level that many social systems operate far from equilibrium.

Nonlinearity

The nonlinearity of human relations is notorious, as illustrated by numerous examples (minor disagreements escalating into major disruptions in relationships; insignificant restrictions causing tantrums; and the assassination of Archduke Ferdinand in Sarajevo provoking the First World War).

Self-Organisation

In rigidly structured social systems where members are centrally controlled, self-organisation is very weak and may not even exist. In contrast, when constituent agents are given some freedom to make autonomous decisions, they will use this freedom to pursue their goals in the presence of disruptive events, thereby engaging in self-organisation.

Social agents differ widely in their abilities to make decisions under conditions of uncertainty, which affects the self-organising capabilities of social systems.

Coevolution

All social systems change over time. These changes are influenced by the interactions of their agents with those of other systems. If we define the environment as the set of all systems with which the system under observation interacts, then the system evolves (changes) through interactions with its environment. Because the system's environment also changes, it is appropriate to use the term coevolution (of the system and its environment) rather than evolution.

Tuning Social Complexity

Social complexity can be tuned.

It can be increased to encourage self-organisation, adaptability, resilience, and sustainability.

It can be decreased to control extreme events.

The following properties can be adjusted.

Connectivity of agents (the number of other agents to which an agent is connected) increases complexity; conversely, higher connectivity reduces complexity.
Autonomy of agents (agents' freedom to choose how to behave) increases complexity; conversely, increasing autonomy increases complexity.

The most effective method is to restrict connectivity by organising a social system as a network of swarms.

Protecting Society from Extreme Global Events

Organising Global Social Systems as a Network of Swarms

Increasing the independence of nations, or friendly groups of nations, by ensuring domestic production of critical goods and services and reducing global connectivity would have numerous positive effects.

Limit the speed of butterfly effects, including the spread of infectious diseases.
Reduce economic dependence on unfriendly nations.
Reduce wasteful use of energy on transporting goods and people.

Employing a Swarm of Agents to Detect Fraud and Errors

Swarms of digital agents can cost-effectively detect and expose fraud and errors, as described in Chap. 2. They could prevent the next global financial crisis, similar to the one in 2008.

Employing a Swarm of Agents to Defend Nations

Swarms of digital agents can cost-effectively deploy defensive shields around nations and large naval vessels. They could prevent unfriendly nations from attacking.

Replace Traditional Software with Swarms of Agents

The mass production of traditional software, like the mass production of cars, is vulnerable.

Software must be continuously updated and improved versions released; yet, we have a large number of legacy programs.
A single error in an update of a globally distributed software product can cause a very costly worldwide disruption.

In contrast, swarms of digital agents continuously self-organise and are therefore ever-changing, coevolving with their environment, which makes them sustainable and error-resistant. They can cost-effectively outperform traditional software in most applications.

Important Features of a Complex Society

Emergent Intelligence

The key difference between social complex systems and biological, physical, or chemical systems lies in the degree of intelligence exhibited by their constituent agents. For these discussions, let us define intelligence as "the capability of making decisions under conditions of uncertainty". Intelligence encompasses motivation, the ability to learn, investigate, communicate, and create. This feature of social systems is important because intelligence empowers members to make choices.

A social system comprising intelligent members, given the appropriate autonomy to negotiate, at every moment, what the most worthy common goals are and how to achieve them, exhibits emergent intelligence greater than the sum of the constituent members' intelligence.

The appropriate degree of agent autonomy depends on agents' intelligence and the complexity of the environment in which the social system is embedded. In that respect, each system is different at any given point in time. We can only say that the appropriate autonomy is always greater than none and less than total.

Emergent Creativity

To survive and prosper in the complex world, it is necessary to continually review goals and invent new ways to achieve agreed-upon aims and objectives. To satisfy this need, members should not merely react to disruptive events; they should be creative, able to anticipate trends, and generate new opportunities. Creative members can be appointed (e.g., research and development staff) or emerge when creativity is required (emergent creativity).

My research indicates that emergent creativity is more promising.

Emergent Leadership

Meeting difficult goals requires more than intelligence and skills; motivation is also needed. A leader is a member and social agent capable of motivating and mobilising other members to undertake difficult tasks, particularly those critical to achieving system goals. Leaders can be appointed or allowed to emerge when leadership is required (emergent leadership).

The idea of emergent leadership is relatively new and untested; yet, it is much more aligned with complexity-tolerant thinking than the current practice of appointing leaders.

Control Versus Self-Organisation

The freedom to exercise choice within social systems is never complete. As a rule, the autonomy of social agents, and thus their freedom of choice, is limited by social conventions and norms, by ethical standards, by rules and regulations set out in social system statutes, and by national and international laws that are enforceable through punishment, which can be severe (expulsion from a school, club, or business and deportation from a country) or very severe (imprisonment and capital punishment).

The purpose of limiting agent autonomy has always been to eliminate or reduce the unpredictability of the emergent behaviour of social systems; in other words, to ensure that the systems behave as closely as possible to how their creators intended. However, efforts to control the system by controlling constituent agents are often self-defeating [2].

The notion that a social system can be controlled by excessively restricting the autonomy of constituent agents warrants careful re-examination. The notion is possible if the system is closed or its environment is stable and free

of disruptive events. Such situations do not occur naturally in the real world but are occasionally imposed artificially (e.g., the Berlin Wall).

When these conditions are not met, i.e., when the system is open and its environment is complex (ever-changing in an unpredictable manner), attempts to control a social system by imposing excessive restrictions on agent autonomy are counterproductive. They prevent agents from responding positively to disruptive events and thus stifle self-organisation, which, over time, leads to system disintegration (remember centrally planned economies?).

Even more importantly, when the perception of the desirable autonomy of agents by those attempting to control a social system and by constituent agents themselves differs significantly, each agent tends to formulate a private (undeclared) set of goals that may not be compatible with the publicly declared social objectives.

Activities aimed at achieving non-declared goals are often conducted covertly, resulting in "deviant" behaviour (infidelity, lying, theft, and murder) and/or in organising resistance to change official goals (rebellions and revolutions).

In social systems with strict centralised control, the unofficial (underground and dissident) behaviour exhibits all features of complexity, including emergence and self-organisation, which ensures its long-term success, as experienced by disciplinarian institutions and totalitarian political regimes.

We have to come to terms with the notion that complex social systems cannot be controlled and learn how to design them to be self-organising.

Self-Organising Geopolitical System

The geopolitical system is the most complex social system, encompassing approximately seven billion people worldwide. Until we find our footing in this new field of research, it is wise to assume that the smallest component of the geopolitical system (its constituent Agent) is a nation or a large international movement. Then we could develop models of geopolitical constellations using complex adaptive systems and test them to assess their adaptability and resilience.

A quick analysis reveals that, although we never had a global dictatorship, we experienced a geopolitical constellation of very low complexity in the twentieth century following World War II, characterised by equilibrium behaviour due to the presence of two powerful political blocs facing each other, armed with nuclear weapons.

We experienced a geopolitical system consisting of two antagonistic agents, both powerful because each possessed weapons capable of destroying life on the whole planet, yet powerless because they could not use these weapons without causing self-destruction.

Except for this period in the twentieth century (the Cold War), throughout recent history, the geopolitical system behaved as a genuine complex system. Whenever a nation is attacked by another, the attacked side manages to self-organise and successfully defend itself.

According to my analysis, in recent history, no nation that initiated a war has won it.

Consider Napoleon's disastrous attempt to conquer Russia. As Napoleon's army approached Moscow, the highly centralised Russian state abandoned the defence of its capital and withdrew its forces, allowing Napoleon to enter the city. But Moscow was empty; its citizens had decided to abandon it. When Napoleon entered the city, they set fire to their own homes and expelled the enemy without suffering any human losses.

A perfect example of self-organisation is self-defence. Napoleon was then chased by partisans and the regular army all the way back to Paris.

Hitler's defeat at Stalingrad followed a similar pattern.

Current geopolitical conflicts are even more complex. Mobile phones and the Internet empower the attacked side to rapidly self-organise into a formidable defence network (guerrilla warfare).

No recent military intervention by one geopolitical agent against another has achieved its goals because of the resilience of defenders created by social complexity.

The technology-driven complexity of the twenty-first-century world calls for new geopolitical behaviour: conflict resolution rather than aggressive intervention.

Adaptive Defence

You don't have to be a visionary to understand that the Swarm AI will play a major role in future military conflicts, if any.

Interlinked swarms of agents can defend digital infrastructure against cyberattacks and national borders against missile attacks.

They can replace drivers, pilots, drone navigators, and naval personnel in a variety of military vehicles, spacecraft, submarines, and ships.

There is no doubt that future wars will be fought between autonomous tanks, drones, aircraft, ships, submarines, and missiles.

The key question will be: whose army of competent Python coders is the best?

Adaptive Healthcare

Current national healthcare systems are, in general, traditional and rely on outdated computer systems.

New initiatives are rare but radical.

Let's consider the remarkable Healthcare Sector Transformation Program of the Kingdom of Saudi Arabia.

The aim is to transform the Kingdom's healthcare system to be more comprehensive and cost-effective, prioritising disease prevention and improving access to healthcare through the introduction of e-health services.

The recently launched SEHA Virtual Hospital is the largest of its kind in the world, connecting over 150 hospitals and more than 30 specialised health services into a distributed, adaptive virtual healthcare system.

Preventive healthcare is based on the idea of providing citizens with wearable sensors whose outputs are monitored, and out-of-range signals trigger the attention of human or AI clinicians.

Virtual NHS

The NHS is a grotesquely large, rigidly structured organisation. At the time of writing, NHS England had a budget of nearly 200 billion and employed 1.5 million staff.

It could be transformed, step by step, into a virtual enterprise, a network of adaptive units (hospitals, GP surgeries, clinics, and independent clinicians), scheduled by swarms of digital agents.

Smart City

Urban and agricultural parts of society may be organised differently, taking into account the distinguishing features of each.

Let's consider an urban settlement, such as a town or a city, and explore how to make it adaptive. An adaptive city is often referred to as a smart city [3].

At present, all vital services for citizens are managed individually, and coordination is conducted intermittently through a series of meetings. As a result, there is considerable resource waste.

Urban environments are highly complex; they are characterised by frequent disruptive events (such as changes in demand, resource failures, or delivery delays caused by accidents, traffic jams, shopping sprees, festivities, tourist seasons, epidemics, etc.).

Negative consequences of these disruptions can be eliminated, or at least reduced, only by rapid rescheduling of the disrupted service without disrupting unaffected services.

No centralised planner could handle this complexity.

In contrast, a smart city comprising a network of cooperating and/or competing adaptive schedulers ensures that constituent services adapt to changes in demand or resource availability.

In addition to a network of adaptive schedulers (Chap. 3) that ensure service adaptability, a smart city features a network of adaptive pattern recognisers (Chap. 4) that dynamically extract knowledge from data.

Let us consider a typical urban settlement governed by a council and managed by the town administration, which is responsible for services to citizens, such as education, social services, roads and transport, waste disposal, economic development, planning, public protection (from crime, fire, the elements, etc.), libraries, environmental health, tourism, leisure and amenities, and the collection of council tax.

Assume that the council has decided to substantially improve each service to citizens while simultaneously reducing the costs of service provision.

The services may be delivered by the town administration or outsourced to private companies, in which case we would be dealing with an ecosystem comprising both public and private sectors.

Requirements Specification

The conventional wisdom is to approach the problem top-down by attempting to specify requirements for the entire town before beginning design work.

Such an approach would be unacceptable in the current political, social, economic, and technological landscape. The dynamics is such that requirements would be obsolete before being completed.

The author advocates an evolutionary design method, as described below.

Evolutionary Design Method

Under conditions of frequently changing requirements, it is advisable to adopt an evolutionary design method, as follows:

Consider all services with the aim of identifying the one whose improvement would yield the greatest value to the client.
Complete a requirements specification for the improvement of the selected service.
Design a component of a smart city that consists solely of the selected service.
Evaluate and update the solution, demonstrating the achieved increase in the client value.
Select the next service and repeat the above steps, ensuring that the newly added service cooperates or competes with the previously improved services, as required.
As the number of improved services increases, continuously monitor and, if necessary, adjust the overall ecosystem design.

In theory, the evolutionary improvement of services should never stop. In practice, contractual arrangements between clients and digital ecosystem designers cannot be indefinite and must be limited to improving one or more services at a time.

A wise client would ensure they have priority access to a skilled team to continuously maintain and improve the new digital ecosystem. And that would represent an extension of the urban ecosystem with additional self-maintenance and self-improvement services.

Concurrent Design Mode

In addition to adopting an evolutionary approach to the design, the ecosystem designers should be organised into three teams working concurrently on requirements specifications, the design, and the commissioning of improved services, with the work timed as follows.

Once the design for improving a service is complete, the requirements specification for the next service is ready for design.
Whenever the commissioning of the improved service is complete, the design for the improvement of the next service is ready for commissioning.

Let's assume that the critical service selected for the improvement is the ambulance service.

Designing Ontology

The design begins with outlining ontology for the ambulance service, which involves:

Selecting object classes (ambulance vehicle, ambulance crew, crew member, ambulance equipment, route, road, hospital, and patient).
Identifying relations (crew belongs to vehicle, equipment belongs to crew, and vehicle follows route).
Defining properties of object classes (for crew member: id, qualifications, and availability).

It is prudent at this stage to write scripts for agents, which should also be stored in the ontology as properties of object classes and made ready for agents to retrieve when a task is assigned.

All policies, rules, and regulations guiding the delivery of services should also be stored in ontology.

Designing Swarm Engine

The swarm engine is the complete software infrastructure that supports the exchange of meaningful messages among potentially hundreds of thousands of agents. The choice of technology is vital, and our recent switch to Python and Microservices has delivered considerable gains: development time has been almost halved!

The swarm is a place where agents negotiate how to allocate resources to demands. The process is rapid and, in principle, not repeatable. The conditions under which negotiations between agents take place often change during the negotiations. In this respect, the swarm is like Heraclitus' river: no agent can enter the same swarm twice.

In the Ambulance Service swarm, the matching of crew members to crews, crews to ambulance vehicles, ambulance vehicles to hospitals, roads to routes, and patients to hospitals is performed through message exchanges between relevant agents.

Adaptability is achieved by matching demands to resources through agent communication rather than computation. For example, if a road to a selected hospital is blocked by excessive traffic, the road agent for that road will immediately notify other road agents of the problem and trigger a wave of renegotiations among affected agents to determine a new route to the hospital. If a new route is too long, agents may negotiate a different hospital.

The key advantage of this type of rescheduling is that parts of the Ambulance Service ecosystem unaffected by the road closure continue to function as though no disruption occurred.

All scheduling decisions made in the swarm must be rapidly conveyed to the real world, the preferred method being messaging to smartphones or directly to relevant "things".

What is described here is only a minute part of the design process, but it is sufficient to demonstrate that agent-based technology offers a considerable advantage over conventional batch-mode optimisers. Additional design details are provided in [2].

Extending the Initial Design

Ambulance scheduling can be readily extended to cover, for example, the scheduling of hospital staff and facilities required for patients transported by ambulance. We require an additional swarm of agents to perform this task. We can then add further swarms of agents to schedule other hospital resources, such as operating theatres. The design should proceed step by step, with each step demonstrated in practice before the next is commissioned.

Important Points

1. Except for dictatorships, dictatorial corporations, and rigid families, social systems are complex systems and, therefore, adaptive.
2. We can design adaptability into society, as illustrated by the design of a smart city.

References

1. Rzevski, G., "The Future is Digital: How will Complexity and AI change our Lives and Work". Springer Nature, 2024.
2. Rzevski, G. "Harnessing the Power of Self-Organisation" International Journal of Design & Nature and Ecodynamics, Volume 11 No 4 (2016), pp. 483–494. ISSN: 1755-7437.
3. Rzevski, G., Sergei Kozhevnikov, Miroslav Svitek. "Smart City as an Urban Ecosystem", 2020 Smart City Symposium Prague (SCSP), https://doi.org/10.1109/SCSP49987.2020.9133849.

Swarm AI Terminology

Adaptive Capable of self-organising with a view to eliminating, or at least minimising, the negative consequences of a disruptive event (e.g., a cancelled order, a failed resource, or an attack by hackers).

Agent An algorithm capable of composing, interpreting, and exchanging messages with other digital or human agents with a view to making deals or resolving conflicts by negotiation.

Autonomy The capability of making decisions under conditions of uncertainty.

Butterfly effect The movement of a butterfly's wing at one end of the world may cause a storm at the other end (a single virus escaped from a laboratory in Wuhan and rapidly became a pandemic).

Class of a domain object The class of an object belonging to the knowledge about the section of the real world under consideration.

Coevolution Two or more systems adapting to one another, each capable of changing over time through mutual influence.

Collective intelligence The intelligence of a swarm is greater than the sum of the intelligence of constituent agents.

Complexity A situation is complex if it consists of interacting participants that are not centrally controlled.

Digital economy Machines perform certain mental tasks better than people.

Digital transformation Introducing digital technologies, particularly AI, into an enterprise to enhance adaptability.

Digital twin A digital twin is a swarm of digital agents that models a section of the real world, with every demand and resource of the real world represented by an agent. The role of a digital twin is to address complex real-world problems that are too difficult for humans to solve.

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2026

G. Rzevski, *Swarm AI*, https://doi.org/10.1007/978-3-032-26262-2

Disruptive event An event that disrupts equilibrium.

Domain knowledge Knowledge of the section of the real world under consideration.

Drift into failure Small errors or frauds accumulate over time and, when the tipping point is reached, cause significant damage (accumulation of small toxic loans caused a worldwide financial crisis).

Emergent property Property of a system that is not attributable to any of the system components; it emerges from agent interaction.

Emergent intelligence Intelligence of a system that is not attributable to any of the system components; it emerges from agent interaction.

Emergent creativity Creativity of a system that is not attributable to any of the system components; it emerges from agent interaction.

Extreme event Rare, unpredictable event that disrupts equilibrium with extreme consequences.

Gaia A hypothesis that the Earth functions as a superorganism, self-organising and exhibiting emergent intelligence.

Industrial economy Machines perform certain manual tasks better than people.

Knowledge base Software where domain knowledge is deposited.

Microservice Software modules that service agents' computational requirements.

Ontology Software where conceptual knowledge is deposited in the form of a semantic network.

Powerful blow An aggressive attempt by one agent to destroy another may cause serious disruptions across the whole world (the invasion of Ukraine caused disruption of vital supply chains and triggered inflation and shortages in Europe, Africa, and elsewhere).

Resilient Capable of resisting attacks, such as hacking and fraud.

Self-organisation The capability of a system to modify its own structure to adapt to changes in its environment without any external intervention.

Semantic network A knowledge structure that uses a graph of nodes and links to represent concepts and the relationships between them. It visually and logically organises information, with nodes standing for concepts (classes of objects) and links representing the connections, such as "is a" or "has a". This structure is used in artificial intelligence to help machines understand relationships and perform tasks like natural language processing, search, and reasoning.

Swarm AI architecture A swarm of agents connected to a knowledge base.

Swarm of agents A group of agents engaged in interaction without being centrally controlled.

Swarm engine Software that supports a swarm of digital agents.