Functional Modelling of Complex Systems

Abstract

Complex socio-technical systems produce behaviour through interactions among human, technical, organisational, informational and environmental elements. Their functional modelling requires a clear purpose and boundary, an account of what each function needs and produces, and a representation of how performance changes over time. This paper presents a practical route from systems thinking to executable functional models. SEIPS supports a structured definition of the work system and its context; FRAM represents functions, aspects, couplings, variability and instantiations; and a governed metadata layer gives functions explicit states and defines how couplings, internal dynamics and external conditions change the system state. The resulting model can support scenario analysis, simulation, assurance, incident investigation and comparison with operational data. Its validity remains conditional on evidence, transparent assumptions, expert review and behavioural testing.

Keywords: *functional modelling; complex socio-technical systems; FRAM; SEIPS; executable models; system assurance.*

Introduction

A system is an open and changing arrangement of interdependent human, technical, organisational, informational and environmental elements whose interactions produce capabilities, behaviour and outcomes over time. Understanding it therefore requires more than an inventory of people, equipment, organisations or procedural steps.

Every system model is constructed for a purpose. The question being asked determines the viewpoint, boundary, operating context and level of detail. Feedback, constraints, resource limitations, delays and local adaptations shape performance, while their interactions can produce outcomes that cannot be explained by examining individual components in isolation.

Functional modelling concentrates on what the system does. It identifies the functions through which the system achieves its purpose, what those functions need and produce, and how their performance affects one another. It must represent both the potential organisation of the system and the different ways in which that organisation may be realised under particular conditions.

This paper develops a practical route from systems thinking to a defensible and executable functional model. The method can be summarised in six stages:

1. Define the modelling purpose, viewpoint, operating context and relevant outcomes.
2. Establish a provisional system boundary and identify its external interfaces.
3. Gather evidence about the people, tasks, technologies, organisation and environment that shape the work.
4. Identify the functions, aspects and potential couplings through which the system produces its capability.
5. Examine credible instantiations with people who understand the work.
6. Add governed metadata, execute the model and test its behaviour against evidence and expert judgement.

The sections that follow explain the foundations of this approach, the use of SEIPS in defining a healthcare work system, the role of FRAM in representing functional dependency and variability, and the additional semantics needed to make the model executable.

1 Evolution of systems thinking and modelling

The development of systems thinking can be understood as a widening of the unit of analysis. Early engineering analysis concentrated mainly on components and direct cause-and-effect relationships. This remains effective where a system can be decomposed without losing the behaviour of interest. It becomes less adequate when outcomes depend upon feedback, adaptation, timing and changing conditions.

General system theory and cybernetics brought organisation and interaction into the analysis. Ludwig von Bertalanffy described living and organisational systems as open systems exchanging matter, energy and information with their environments. Norbert Wiener and W. Ross Ashby developed related accounts of communication, feedback, control and regulation. System behaviour could now be understood as a product of both the elements and the relationships that organised them (von Bertalanffy, 1968; Wiener, 1948; Ashby, 1956).

Systems engineering developed during the same period in response to increasingly complex telecommunications, defence and aerospace projects. It brought together objectives, requirements, interfaces, human limitations, technical disciplines and lifecycle considerations. Its early practice was a pragmatic way of coordinating complexity; later standards formalised the translation of needs and constraints into designed system solutions (SEBoK Editorial Board, n.d.).

The socio-technical tradition widened the focus beyond engineered hardware. Trist and Bamforth's study of British coal mining showed how a technically efficient production arrangement could disrupt established working relationships and impair the work system as a whole. The social and technical arrangements had to be understood together (Trist and Bamforth, 1951).

Soft Systems Methodology addressed situations in which purposes, boundaries and even the nature of the problem were contested. Checkland treated a system model as a purposeful intellectual construct used to explore a real-world situation from a stated worldview. Different participants could therefore produce different coherent accounts of the same situation, making modelling an organised process of enquiry and learning (Checkland, 2000). System dynamics added explicit representations of accumulation, flow, feedback and delay, allowing changes in behaviour to be explored through time (Forrester, 1961).

Model-Based Systems Engineering, or MBSE, developed as models began to carry an increasing share of the formal system definition. INCOSE describes MBSE as the formalised use of modelling to support requirements, design, analysis, verification and validation across the system lifecycle. Its value lies in consistency, traceability and a shared basis for engineering decisions (INCOSE, n.d.).

Healthcare systems thinking developed a particularly useful work-system perspective. The Systems Engineering Initiative for Patient Safety, or SEIPS, connected people, tasks, tools and technologies, organisational arrangements and the working environment with care processes and outcomes. SEIPS 2.0 gave greater attention to configuration, engagement and adaptation, while SEIPS 3.0 extended the framework across the patient journey and organisational boundaries (Carayon et al., 2006; Holden et al., 2013; Carayon et al., 2020).

Resilience engineering and the Functional Resonance Analysis Method, or FRAM, brought performance variability and emerging functional relationships into the foreground. FRAM describes the functions required for an activity to succeed, the conditions upon which they depend, and the ways in which normal variation may combine across the system (Hollnagel, 2012).

These traditions remain complementary. Systems thinking supplies the concepts of purpose, boundary, feedback and emergence. Systems engineering contributes disciplined definition and lifecycle assurance.

MBSE provides formal representation and traceability. SEIPS offers an established socio-technical account of healthcare work systems. FRAM represents functional dependency, variability and instantiation. Together they provide the foundations for a model that can describe how a complex system works and how its behaviour changes.

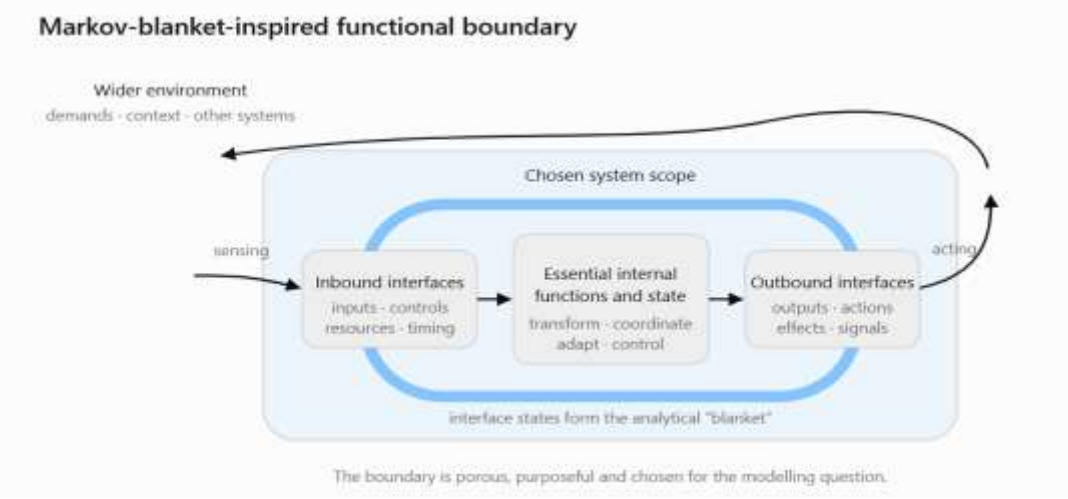
2 First define the system

The systems modeller needs a simple opening rule: **first define the system**. Before identifying functions or drawing a model, the analyst must establish what the system does, what it needs and how it is connected to the wider environment.

What the system does establishes its purpose, capabilities and relevant outcomes. The definition should state the question being addressed, the viewpoint from which the system is examined and the operating circumstances that matter. A hospital, aircraft or process plant rarely has a single uncontested purpose. Participants may understand priorities differently, while actual behaviour may reveal pressures and practical goals that do not appear in the formal description.

What the system needs identifies the conditions that sustain its performance. These may include people, equipment, information, materials, authority, procedures, capacity, prior conditions, time and feedback. At this stage they should be described in the language of the work. Formal modelling categories can be applied once the system has been understood on its own terms.

The system also exchanges demands, resources, information, constraints and disturbances with its environment. Its boundary is therefore analytical and permeable. It is located at the interfaces through which external influences enter and system effects leave. The boundary should contain the essential internal functions needed to explain the capability of interest. Other systems remain part of the context unless their internal operation must be understood to answer the modelling question.



As a boundary-design heuristic, this can be compared with a **Markov-blanket-inspired functional boundary**. In its formal probabilistic sense, a Markov blanket comprises the interface states that conditionally separate the internal states of a system from its external states (Pearl, 1988):

$$I \perp E \mid B$$

Inbound states convey environmental influence into the system; outbound states convey the system's effects into its environment. A functional boundary does not automatically satisfy this mathematical condition, and

the analogy does not establish that a functional model has a formal Markov blanket. It provides a disciplined way of identifying the exchanges through which the focal system and its environment affect one another.

Using SEIPS to define a healthcare work system

SEIPS provides an established and widely used foundation for defining healthcare work systems. The original model describes five interacting work-system components: people, tasks, tools and technologies, organisational conditions and the physical environment (Carayon et al., 2006). SEIPS 2.0 makes the external environment explicit and gives greater attention to configuration, engagement and adaptation (Holden et al., 2013). For practical investigation, these can be examined under six broad headings: people, tasks, tools and technologies, organisation, internal environment and external environment (NHS England, 2022). This six-element SEIPS work-system lens helps the analyst examine the setting in which care is produced and reduces the risk that important influences will be overlooked.

Within a functional modelling process, SEIPS supplies structured evidence and context. It helps establish what the system contains, what conditions shape its performance, and how it is linked to the wider environment. Functional System Modelling then asks what this combined work system must accomplish and how its activities depend upon one another. The six elements are interpreted rather than converted mechanically into functions. A person may perform or coordinate a function; a technology may support, constrain or perform one; an organisational arrangement may supply control or resources; and an external service may remain a boundary condition or require modelling in greater detail.

The relationship is cumulative. SEIPS contributes a mature socio-technical understanding of healthcare work and an extensive body of empirical application. Functional modelling adds an explicit representation of how the work system produces its behaviour. The SEIPS-informed evidence, its provenance and the analyst's decisions should remain available as the functional model is constructed and reviewed.

A completed System Definition should record the modelling purpose, viewpoint, intended and relevant unintended outcomes, operating context, essential internal functions, external systems, inbound and outbound interfaces, feedback relationships, exclusions and assumptions. Its boundary remains provisional. If an external influence cannot be represented adequately as an interface condition, the boundary should expand to include the function that produces it.

The definition is ready to support modelling when the included functions appear sufficient to explain the stated capability, important external influences and effects cross through identifiable interfaces, and feedback, control and adaptation have been considered. This gives the model a defensible scope while allowing revision as understanding develops.

3 What the system does and how it behaves

Once the system has been defined, attention moves from its contents to its activity. People, equipment, information and organisational structures provide the means through which it operates. The functional question concerns what they accomplish together, what each activity produces and what conditions it requires.

Purpose, capability and outcome describe different things. Purpose states why the system is being considered. Capability describes what it can achieve under stated conditions. An outcome is what results from a particular instantiation. Keeping these terms separate helps the modeller distinguish intended design from observed performance.

A function is a purposeful activity contributing to a system capability or outcome. It acts upon or responds to something, performs a transformation or change, and produces something needed elsewhere. Its performance may depend upon people, technology, information, resources, prior conditions, authority and time. These dependencies connect the function to the wider system.

At a chosen level of analysis, a function can be treated as a black box. The modeller identifies what enters, what the function does and what it produces, without representing every internal detail. When greater explanation is required, the box can be opened and modelled as a subsystem containing its own interacting functions. A coherent group of detailed functions can also be represented as a higher-level function. Functional modelling is therefore recursive; its resolution is determined by the modelling purpose and the behaviour that must be explained.

The functions describe the potential organisation of the system. Behaviour emerges when they operate under particular conditions. Outputs from one function may enable, constrain or modify others. Functions can operate sequentially or concurrently, while feedback may influence later decisions, priorities and resource availability. External events introduce further demands and disturbances through the system boundary.

The same functional organisation can consequently produce different behaviours. Activities occur at different times and with varying resources and effectiveness. People adapt, redistribute effort, defer work and create workarounds. A realised pattern of activity under specified conditions is an **instantiation** of the system. It describes one behaviour from the range of possibilities represented by the model. A practical method is needed to describe functions, interdependencies and instantiations consistently. FRAM offers an established and accessible starting point.

4 FRAM as an accessible functional modelling framework

FRAM represents complex socio-technical systems as sets of interacting and interdependent functions. It begins with the activities needed for the system to achieve its purpose rather than with a component inventory, failure sequence or predetermined causal chain (Hollnagel, 2012).

A FRAM function may be performed by a person, team, technical system, organisation or combination of these. It is characterised through six aspects:

Aspect	Meaning
Input	What initiates the function or is transformed by it
Output	What the function produces
Precondition	What must already be true
Resource	What the function needs while it is performed
Control	What guides, supervises or constrains it
Time	The temporal conditions affecting its performance

These aspects organise what each function needs and produces. They also expose functional dependence. A coupling is formed when the Output of one function supplies an Input, Precondition, Resource, Control or Time condition for another. Outputs are therefore the active sources of coupling; the other five aspects receive their effects.

The resulting structure need not follow a fixed process sequence. Functions may operate concurrently, several upstream outputs may contribute to one activity, and one output may support several downstream functions. Feedback creates recurring activity and continuous cycles. The same framework can therefore describe a once-through passage from entry to exit, a continuously operating system, or a combination of sequential, concurrent and recurrent behaviour.

The distinction between the FRAM model and an instantiation is important. The model records the functions and couplings that may be relevant. An instantiation records those that become active during a particular event, period or set of conditions. Different instantiations of the same model can represent routine work, adaptation under pressure, disruption and recovery.

FRAM also recognises that everyday performance varies. Timing, precision, completeness and effectiveness change as people and technology respond to local demands and available resources. The system may absorb or compensate for this variability, while some combinations may amplify it. Functional resonance describes the emerging effect of these interactions across the system.

At this stage, FRAM provides a structured qualitative model of what the system does and how its functions may interact. The hexagons and couplings do not by themselves define a simulation. Computational execution requires explicit operational semantics, which can be supplied through governed metadata without altering the accepted functional structure.

5 From a functional map to an executable model

A functional diagram shows the system's organisation and its potential interactions. An executable model must also represent the changing state of every function and the rules through which states influence one another. Functions become stateful entities, while couplings become conditional mechanisms for state change. They may be drawn as nodes and arrows, but their computational meaning extends beyond a directed graph.

Each function has a state at a particular time. Depending on the modelling purpose, this may include availability, readiness, activity, effectiveness, resource level, current output, tolerance and uncertainty. A function state need not be a single value; it can be a typed record or vector containing the variables needed for the modelling purpose. The complete system condition at time t can be represented as an array of function states:

$$X(t) = \{x_1(t), x_2(t), \dots, x_n(t)\}$$

This is an instantaneous functional picture of the system. It shows which functions are available, active, constrained, degraded or waiting; what they need and produce; and how effectively they are contributing to the system's purpose.

A coupling describes a potential route of influence. It becomes an actual interaction when the source function produces the relevant output and the receiving conditions are satisfied. The transmitted output may then alter the receiving function's readiness, resources, control, timing or effectiveness. Potential couplings describe what could interact; realised coupling events record what did interact during an execution.

Complexity does not require unexplained all-to-all interaction. Influence should pass through an explicit coupling, shared resource, control mechanism or environmental condition. This permits late-binding and changing configurations while preserving traceability.

Function states can also change without a direct coupling event. Equipment degrades, people become fatigued and recover, stocks are depleted and replenished, demand accumulates, queues grow, information becomes outdated and deadlines expire. These internal dynamics must be represented alongside the interactions between functions.

Metadata makes these behaviours executable. It may define initial conditions, units, aspect requirements, transition logic, timing, delays, thresholds, tolerances, coupling strength, resource consumption, accumulation, degradation, recovery and uncertainty. It can also specify the conditions that begin, repeat or terminate an execution. The accepted FRAM model remains the structural authority; the metadata supplies the governed operational semantics.

The evolution of the system state can be expressed generally as:

$$X(t + \Delta t) = \Phi\big(X(t), U(t), \Theta, V(t)\big)$$

Here, $U(t)$ represents influences entering from the environment, Θ contains the governed parameters and transition rules, and $V(t)$ represents variability or uncertainty. The transition function Φ combines internal change with the effects of realised interactions to produce the next system state.

Execution produces a trajectory of successive state maps:

$$X(t_0), X(t_1), X(t_2), \dots, X(t_k)$$

This trajectory is the dynamic instantiation of the functional model. A once-through process develops towards a defined exit or outcome. A continuously operating system repeats and cycles as its conditions change.

The model is also experimentally manipulable. Initial conditions, demands, resources, controls and variability parameters can be changed while the governed structure is retained. Scenarios can be compared, and repeated runs can explore sensitivity, uncertainty and the distribution of possible trajectories. Operational records can provide further instantiations against which model behaviour is compared.

Every transition should retain the previous state, the interaction or internal rule that produced the change, the metadata applied and the resulting state. This makes the execution explainable. The analyst can examine what the system was doing, how its functions were interacting and why a particular outcome arose within the boundary and assumptions of the model.

Executability alone does not establish validity. The model must also be structurally consistent, functionally credible, behaviourally plausible and supported by appropriate evidence. Its behaviour should be reviewed by people who understand the work and, where possible, tested against observed data.

What executability requires

The general state equation describes an execution architecture. A working model also needs explicit definitions for the following elements:

Requirement	Operational meaning
State schema	Named and typed variables for each function, with units, ranges and constraints
Initial conditions	Starting values, their provenance and any uncertainty attached to them
Activation logic	The aspect conditions that make a function eligible to act, including any required combinations
Scheduling	Event-driven or time-stepped execution, together with delays, durations and ordering rules
Coupling semantics	The way a realised output changes a receiving aspect or function state
Transition rules	State changes caused by execution, accumulation, depletion, degradation and recovery
Uncertainty	Distributions, correlations, sampling rules and reproducible random seeds where variation is modelled
Termination and outcomes	The conditions that stop, repeat or continue an execution and the outcomes calculated from it
Provenance and versioning	The model, metadata, evidence and software versions needed to reproduce a result

These definitions turn a functional description into an algorithm that can be inspected, executed and repeated. They also provide the basis for determining whether the software has implemented the intended model correctly.

Verification calibration and validation

Verification, calibration and validation address different questions. **Verification** checks whether the encoded functions, couplings, units and transition rules operate as specified. It includes schema checks, coupling consistency, dimensional checks, deterministic replay and tests of known boundary cases.

Calibration estimates or adjusts parameters from observations, experimental evidence or defensible expert judgement. It should record the data used, the method of estimation and the uncertainty that remains.

Validation asks whether the model is an adequate representation for its stated purpose. It examines the credibility of the boundary and functional structure, compares behaviour with observed or independently assessed cases, and tests whether relevant participants recognise the resulting instantiations as plausible. Validation is therefore conditional on purpose, context and evidence. A model that is adequate for training or comparative scenario analysis may not be adequate for operational prediction.

Illustrative example

Consider a hospital medication-administration system whose purpose is to deliver the prescribed medicine to the correct patient, in the required form and at the intended time, and then to observe the response. A suitable boundary might begin with an authorised prescription and end with the recorded administration and clinical observation. Pharmacy procurement, hospital staffing policy and the patient's wider care pathway can remain external unless their internal behaviour is needed to answer the modelling question.

Candidate functions could include authorising the prescription, verifying it, obtaining the medicine, preparing the dose, confirming the patient and current readiness, administering the medicine, and monitoring the response. A verified prescription can supply an Input or Control to preparation; available

medicine can supply a Resource; a prepared dose can provide the Input to administration; confirmed identity and readiness can provide Preconditions; and the observed response can feed back into clinical review.

In an execution, the medicine may initially be unavailable, leaving preparation waiting for a Resource. A delayed supply can narrow the available Time condition, while workload may affect the effectiveness and duration of verification. When the relevant outputs and receiving conditions are present, the couplings become realised events and the model records the resulting changes in function state. Repeated scenarios can then explore how supply delay, workload or recovery measures alter the functional trajectory. This example illustrates the method; it is not a validated clinical model or a prescription for practice.

6 Implications applicability and safe operation

Functional system modelling makes systems thinking operational. It turns an appreciation of complexity into an explicit account of what a system does, what its functions need, how they interact and how their collective state changes over time.

This is particularly relevant to contemporary systems. Healthcare, aviation, transport, energy, process industries and public infrastructure depend upon changing combinations of people, software, automated equipment, organisations and external services. Their boundaries are permeable and their performance is shaped by pressures that cross conventional technical and organisational divisions. Artificial intelligence introduces further functions that may be adaptive, probabilistic and only partly visible to the people working with them.

The approach can be applied throughout the system lifecycle. During conception and design it helps identify the functions and dependencies needed to deliver a capability. During verification and assurance it tests whether necessary controls and resources have been provided. During operation it can show the current configuration and where performance margins may be narrowing. The same model can support incident investigation, organisational learning, training, scenario exploration and the evaluation of proposed changes.

Its contribution to safe operation is significant because safety is produced continuously through interacting functions. Some are technical; others involve observation, interpretation, communication, coordination, adaptation and recovery. A stateful model can show whether these functions are available and effective, what they are waiting for, and how changes in one part of the system are affecting others. This supports earlier recognition of developing vulnerability.

Systematic experimentation extends this value. Changes in demand, staffing, equipment availability, environmental conditions, controls or operating strategy can be introduced as scenarios and traced through successive states. Sensitivity analysis and repeated simulation can reveal which assumptions, resources and interactions exert the greatest influence. Similar outcomes can also be shown to arise through different functional pathways, an important consideration when interpreting apparently stable performance.

Simulation results describe conditional consequences within the model. They should not be presented as operational predictions unless the model has been calibrated and validated for that purpose and the relevant uncertainty is reported.

When operational data are available, observed conditions can be compared with the model. The resulting representation may form the functional layer of a digital twin, showing differences between expected and actual states. This can help distinguish effective adaptation from accumulating pressure or emerging loss of control, while providing a shared basis for discussion among participants with different operational perspectives.

Agent-supported modelling can make the approach more accessible by guiding users through system definition, evidence gathering, function identification, coupling and validation. The agent can propose, question and explain; the analyst retains authority over boundaries, assumptions and acceptance. Versioned models, deterministic execution and traceable evidence provide the governance needed to prevent plausible suggestions from becoming unexamined system facts.

Every model remains a purposeful abstraction. Its usefulness depends upon the appropriateness of its boundary, the quality of its functional description, the validity of its metadata and the evidence supporting its assumptions. Numerical precision cannot repair a misconceived model. Structural verification, behavioural testing, comparison with operational evidence and expert review remain essential.

Functional system modelling therefore provides a common framework for understanding, designing, operating and learning from complex systems. It connects established systems thinking with formal modelling, quantitative experimentation and operational evidence. The system becomes visible as a changing configuration of interacting functions, providing a tangible account of how it is doing what it does, how that performance is changing and where intervention may strengthen safe and resilient operation.

Selected references

- Ashby, W. R. (1956). *An Introduction to Cybernetics*. Chapman & Hall.
- Bertalanffy, L. von. (1968). *General System Theory: Foundations, Development, Applications*. George Braziller.
- Carayon, P. et al. (2006). Work system design for patient safety: the SEIPS model. *Quality and Safety in Health Care*, 15(Suppl 1), i50-i58. https://doi.org/10.1136/qshc.2005.015842
- Carayon, P. et al. (2020). SEIPS 3.0: Human-centered design of the patient journey for patient safety. *Applied Ergonomics*, 84, 103033. https://doi.org/10.1016/j.apergo.2019.103033
- Checkland, P. (2000). Soft Systems Methodology: A thirty-year retrospective. *Systems Research and Behavioral Science*, 17, S11-S58.
- Forrester, J. W. (1961). *Industrial Dynamics*. MIT Press.
- Holden, R. J. et al. (2013). SEIPS 2.0: A human factors framework for studying and improving the work of healthcare professionals and patients. *Ergonomics*, 56(11), 1669-1686. https://doi.org/10.1080/00140139.2013.838643
- Hollnagel, E. (2012). *FRAM: The Functional Resonance Analysis Method*. Ashgate.
- INCOSE. (n.d.). Model-Based Systems Engineering Initiative. https://www.incose.org/group/mbse-initiative/
- NHS England. (2022). *SEIPS quick reference guide and work system explorer*. https://www.england.nhs.uk/wp-content/uploads/2022/08/B1465-SEIPS-quick-reference-and-work-system-explorer-v1-FINAL.pdf
- Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann.
- SEBoK Editorial Board. (n.d.). A brief history of systems engineering. *Guide to the Systems Engineering Body of Knowledge*. https://sebokwiki.org/wiki/A_Brief_History_of_Systems_Engineering
- Trist, E. L. and Bamforth, K. W. (1951). Some social and psychological consequences of the longwall method of coal-getting. *Human Relations*, 4(1), 3-38. https://doi.org/10.1177/001872675100400101
- Wiener, N. (1948). *Cybernetics: Or Control and Communication in the Animal and the Machine*. MIT Press.

David Slater 12th September 2026