EuropeanCity2

Funded by the European Union

EuropeanCity2	
Project number:	**101178170**
Project name:	European City Squared: Computational Social Science Simulation for Democracy
Topic:	HORIZON-CL2-2024-DEMOCRACY-01-06
Type of action:	HORIZON-RIA
Starting date of action:	01/01/2025
Project duration:	36 months
Project end date:	31/12/2027
Deliverable number:	D4.3
Deliverable title:	Overview, Design concepts, and Details (ODD)
EC document version:	Ver1
WP number:	WP4
Lead beneficiary:	1-AU
Main author(s):	Iza Romanowska, AU
Internal reviewers:	Minsu Jang (AU), Jan Lorenz, (CUB), Luis Razo Bravo (EISM), Nikola Grunchevski (HYB), Eliahu Cohen (BIU), Kristoffer Knielbo (AU)
Nature of deliverable:	R
Dissemination level:	PU
Delivery date from Annex 1:	M18
Actual delivery date:	M18

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Executive Agency (REA). Neither the European Union nor European Research Executive Agency (REA) can be held responsible for them.

1. Purpose and Patterns

1.1 Purpose

EuropeanCity2 is a multi-model simulation framework designed to investigate the dynamics of democratic systems across several scales of analysis from individual-level cognition and preference formation through to collective electoral outcomes and institutional resilience under exogenous shocks. The framework does not intend to create a single monolithic model. Instead, it is a constellation of seven interoperable submodels, each addressing a distinct mechanism or scale of democratic behaviour that build upon and feed into each other (Fig.1). These submodels are developed independently by specialist teams and are designed to be integrated selectively, with the nature and tightness of coupling varying across research questions and experimental designs. The constellation architecture also reflects the state of the field: there is currently no consensus theoretical framework for a complete model of democratic behaviour from cognition to collective outcome. The EuropeanCity2 framework is designed to produce that framework iteratively, through selective integration of well-validated submodels.

The overarching goal is to serve as a testbed for theories of democratic behaviour drawn from Social Choice Theory, cognitive science (via Active Inference), and quantum information theory, while remaining empirically grounded through data from real-world participatory governance processes in Aarhus, Denmark. The framework supports both abstract, parameter-space exploration and scenario-based replication of specific governance episodes (participatory budgeting, citizen assemblies, quadratic voting pilots, and more).

A central design principle is modularity with interoperability: submodels expose well-defined input/output interfaces, allowing them to be swapped, composed, or run in isolation depending on the research question. This architecture reflects the project's ambition to bridge theoretical paradigms: classical social choice, Active Inference, and quantum voting with just enough alignment to ensure the models can talk to each other but without presupposing a single unified cognitive or normative framework.

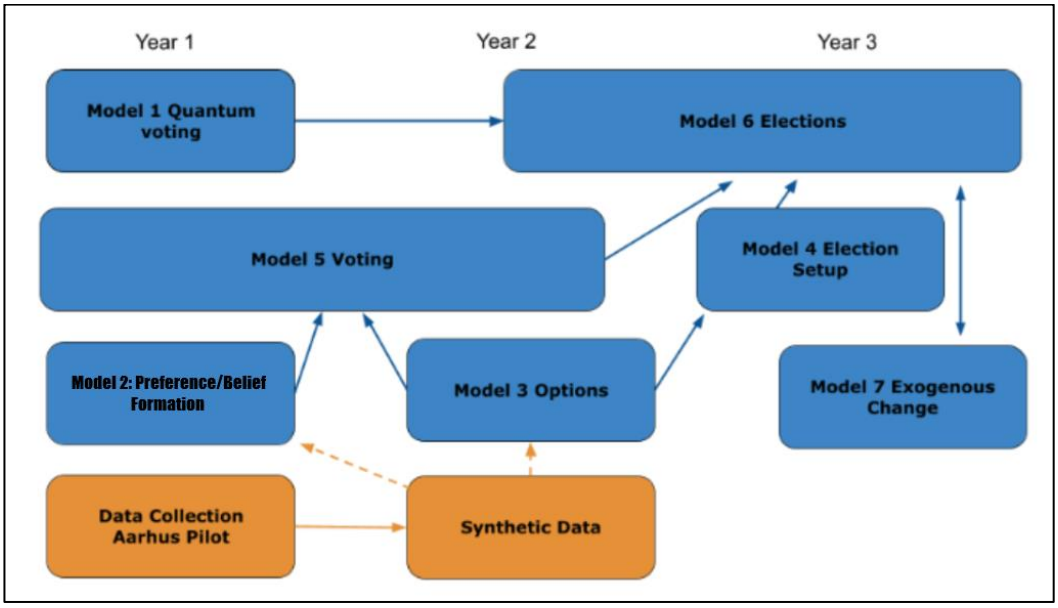

Figure 1. *The overall architecture of EC2, including 7 interoperable models and data layers.*

1.2 Research Questions

The simulation framework is designed to address three clusters of research questions:

Voting Systems and Aggregation Mechanisms

- How do different voting and aggregation methods affect collective outcomes and voter-level satisfaction under varying preference distributions? What mechanisms maximise aggregated utility under varying fairness, efficiency, and equity criteria?
- Under what conditions do different voting mechanisms balance trade-offs between fairness, efficiency, robustness, and voter-level satisfaction? How can preference distributions benchmark voting-method performance under differing scenarios of information availability?
- How do voting mechanisms mediate the tension between individual self-interest and aggregate social welfare? Under what conditions can they preserve representative and fair outcomes despite heterogeneous voter preferences?
- How does quantum-inspired quadratic voting perform relative to classical quadratic and one-person-one-vote systems? Can quantum-inspired voting mechanisms reduce strategic distortion compared to classical models?

Dynamics of Democratic Behaviour and Polarisation

- What feedback loops or learning mechanisms promote voter adaptation, engagement, and consensus, and reduce polarisation?
- How do voters shift between moderate and fringe positions? What is the interplay between individual psychological processes and societal influence dynamics?
- What types of generative models for preference formation produce the most accurate predictions of voter behaviour?
- How do exogenous events influence electoral dynamics, and can simulations identify tipping points in preference and polarisation transitions?

Methodological questions

- How do data biases in LLM-generated preference profiles affect the validity of simulated electoral outcomes?
- Is it possible to integrate quantum computational models, analyses, or output into classical agent-based modelling?

1.3 Patterns Used for Evaluation

Consistent with pattern-oriented modelling (POM), the following empirical and theoretical patterns are used as evaluation criteria:

- Empirical: vote-share distributions from Danish municipal and national elections (Aarhus 2021, 2025); preference distributions from the 2026 voter survey and the DR Kandidattest. Data from 2026-2027 experiments.
- Theoretical: median voter theorem behaviour under plurality rules; Arrow's impossibility conditions under classical aggregation; Duvergerian mechanical and psychological effects of electoral systems on strategic voting and coordination; emergence of strategic voting under information scarcity; proportionality and representation-quality expectations under proportional representation systems.
- Behavioural: polarisation dynamics consistent with empirical literature; magnitude and decay of exogenous shock effects on preference distributions.

2. Entities, State Variables, and Scales

2.1 Entities and State Variables

The EuropeanCity2 framework comprises seven submodels operating over shared core entities (Fig. 1). The entities described below constitute the common ontology of the framework; specific submodels instantiate subsets of these entities and add model-specific state variables. This document specifies the complete target ontology. Successive implementation versions (v1, v2, v3, ...) realise it progressively, so any given build may instantiate only a subset of the variables defined here. Table 1 provides a mapping of entities to submodels.

Entity	M1 QV	M2 Pref/ Belief	M3 Options	M4 Setup	M5 Voting	M6 Election	M7 Exogen.
Voter	✓	✓	partial	–	✓	✓	✓
Group	–	✓	✓	–	✓	partial	✓
Proposal	–	–	✓	✓	✓	✓	–
Election	✓	–	–	✓	✓	✓	partial
Event	–	✓	–	–	–	partial	✓
Polis	partial	partial	partial	✓	✓	✓	✓

Table 1. *Entity–submodel mapping. A tick indicates that the entity is active in the submodel.*

For each entity, the following variables are common to all models. Additional variables specific to submodels are discussed in section 7.

2.2.1 Voter

The Voter is the primary autonomous decision-making entity, representing an individual within the electoral system. Voters are the locus of preference formation, belief updating, and voting behaviour. In the EC2 deliverable D4.1 class vocabulary this entity is the Agent class; Voter is the political-framing name used throughout this document, and the two refer to the same entity.

Variable	Type	Description
preferences	Multi-dim. distribution over policy positions; concrete representation is architecture-specific (see §7.2)	Value system and decision priorities across policy dimensions (e.g., economy, ecology, social policy); concrete form is architecture-specific (see §7.2)
beliefs	Multi-dim. probability distribution	Voter's model of the world; factual understanding of political and social conditions
memberOf	Set of Group IDs	Groups to which the voter belongs; enables group-based social influence
connectedTo	Set of Voter IDs	Social network neighbourhood; governs information flow and peer influence
attributes	Key–value map	Individual sociodemographic characteristics (age, location, employment); defined per experiment

indGravitas	Scalar (float)	Individual influence weight; source is configurable (e.g. earned in-simulation, quantum-derived via M1, or empirically assigned)
trust	Float (0–1)	This voter's trust in electoral outcomes (the per-voter term TrustIndex averages).
perceivedFairness	Float (0–1)	This voter's perception of whether the process was fair (the per-voter term PerceivedFairnessScore averages).
satisfaction	Float (0–1)	Whether this voter personally got the outcome they wanted.
voted	Boolean	Whether this voter cast a ballot in the most recent election.

Table 2. *Voter state variables.*

Voter baseline methods: perceive(), learn(), forecast(), updatePreferences(), selectAction(), initiateComms(), broadcast(), receive(), collapsePreferences(), vote().

2.2.2 Group

A Group is a collection of Voters united by shared attributes, ideology, organisational structure, or functional role (e.g., political party, demographic cohort, media outlet).

Variable	Type	Description
attributes	Key–value map	Collective identity characteristics (ideology, purpose, function); defined per experiment
members	List of Voter IDs	Members belonging to this group
aggregations	Statistical summary	Derived statistics from member attributes (mean preferences, polarisation index, etc.)
trust	Float (0–1)	Mean trust among this group's members.
perceivedFairness	Float (0–1)	Mean perceived fairness among this group's members.
satisfaction	Float (0–1)	Mean satisfaction among this group's members.
turnout	Float (0–1)	Fraction of this group's members who voted.
gravitas	Scalar (float)	Collective influence metric governing the group's impact scope and strength

Table 3. *Group state variables.*

2.2.3 Proposal

A Proposal represents a policy option, candidate, or ballot measure that voters evaluate and vote upon. Proposals are generated exogenously from empirical data or, positively assessed in pilot testing, endogenously via LLMs.

Variable	Type	Description
attributes	Key–value map	Policy parameters, budget implications, scope, and ideological positioning; defined per experiment

Table 4. *Proposal state variables.*

2.2.4 Election

An Election is a decision event in which Voters express preferences over Proposals through a configurable voting method and electoral system, set independently, with a swappable vote decider. Classical and quantum-inspired Quadratic voting are the main voting methods tested.

Variable	Type	Description
date	Integer (timestep)	Simulation time step at which the election occurs
votingMethod	Enum	The ballot mechanic (how a voter marks a ballot): SingleChoice, Quadratic (v1); ranked-choice, approval, etc. in later phases.
electoralSystem	Enum	The aggregation rule (how marked ballots become a result): OpenListProportional, Plurality (v1). Independent of the voting method; either can pair with either.
voteDecider	Enum	The swappable per-voter rule that converts voter's per-voter probabilistic result into a ballot or abstention: EciSampled (default, keeps the sampled vote), ThresholdAbstain (abstains when the top probability is below threshold).
stages	Integer	Number of phases in the electoral process (e.g., primary + general)
alternatives	List of Proposal IDs	Available ballot options
goalType	Enum	Named goal: flat (default), consensus, condorcet, min-polarisation. This maps to a backwardGoal penalty.
backwardGoal	Distribution over outcomes (or a penalty function penalty(x) that induces one)	The objective ⟨φ| that the collapse is conditioned on: (default is no goal). This is collective, not per-voter.

Table 5. *Election state variables.*

2.2.5 Event

An Event is an internal or external input that modifies voter beliefs, preferences, or behaviours. Events include election outcomes, policy announcements, scandals, and environmental shocks. Elections are a special case of Event.

Variable	Type	Description
date	Integer (timestep)	Simulation time step at which the event occurs

targetGroup	Group ID or 'ALL'	Scope of influence; which voters are affected
impact	Scalar or vector (float)	Magnitude and direction of influence on voter belief/preference states

Table 6. *Event state variables.*

2.2.6 Polis

The Polis is the global environment: the formal political system encompassing all voters, groups, proposals, and elections. It functions as the simulation orchestrator and outcome aggregator.

Variable	Type	Description
NVoters	Integer	Total number of voters in the simulation
eventSchedule	List of (timestep, Event)	Ordered schedule of all events including elections
time	Integer	Current simulation timestep
PerceivedFairnessScore	Float (0–1)	Mean voter-level perception of election fairness
TrustIndex	Float (0–1)	Mean utility-weighted trust in electoral outcomes
PolarisationIndex	Float	System-level dispersion of post-election preferences (the spread the min-polarisation goal targets).
RepresentationQuality	Float (0–1)	Objective closeness of the elected outcome to the electorate's aggregate preferences; distinct from perceived fairness.
ParticipationRate	Float (0–1)	Fraction of eligible voters who cast a ballot.
MeanSatisfaction	Float (0–1)	Mean voter-level satisfaction; distinct from PerceivedFairnessScore (was the process fair?) and RepresentationQuality (objective match of result to preferences).
aggregations	Key–value map	Extensible bag for any further system-level metrics not given their own row above.

Table 7. *Polis state variables.*

2.3 Scales

Different submodels work at different spatio-temporal scales.

Dimension	Abstract Scenarios	Empirical Scenarios (Aarhus Pilot)
Temporal unit	Abstract timestep	Calibrated at between seconds for cognitive scenarios to days/weeks of campaign cycle in election scenarios
Temporal extent	Configurable; typically 1–3 election cycles	Municipal election cycle (4 years); national cycle (4 years)

Spatial unit	None (aspatial network)	Aarhus municipality, neighbour-level spatial aggregation
Population size	100–100,000 voters (parameter sweep)	Calibrated to electorate size (~180,000 Aarhus voters)
Policy domains	2–10 abstract domains	15 domains derived from Kandidattest and voter survey questions

Table 8. *Temporal and spatial scales.*

3. Process Overview and Scheduling

3.1 Overall Simulation Process

The EuropeanCity2 framework does not implement a single unified simulation loop. Each submodel is developed and executed independently, and integration occurs through defined I/O exchanges at specified coupling points. The three labels used below (pre-election, election, and post-election) group these coupling points; they are a logical grouping, not a strict temporal pipeline. Chronologically, the formal election setup comes first, at campaign initialisation: the option set (M3) and the voting method, electoral system, and vote decider (M4) are fixed before any preferences evolve or votes are cast. Preference formation (M2, with exogenous shocks via M7) then runs over the pre-election period; individual vote-casting (M5) follows once the ballot opens; and aggregation with feedback (M6) closes the cycle. M1 is an offline quantum-theoretic component that supplies the two-boundary form of the aggregation operator and can exchange influence weights and/or a goal-constraint with the classical model; its exact role and direction are configurable. This sequence corresponds to Phase 0 (setup), Phase 1 (the preference loop), and Phase 2 (vote-casting and aggregation) in the D6.3 implementation.

The framework numbers each of these steps as a submodel (M1–M7), but M3 (option generation) and M4 (election design) are primarily configuration rather than dynamic processes; the recurring per-cycle work is done by M2, M5, and M6. Because M3 and M4 are largely setup, they may in a future revision be folded into initialisation and removed as separate submodels.

Post-election stage encompasses vote aggregation, outcome evaluation, and feedback. Model 6 (Election) aggregates submitted votes using the configured mechanism, computes fairness and trust metrics, and broadcasts results back to voters. This broadcast re-enters the pre-election stage of the next cycle: election outcomes are perceived as Events by voters, triggering a new round of belief updating via M2. This feedback loop — from M6 outcome broadcast back into M2 — is the primary mechanism by which elections shape subsequent political behaviour in the simulation.

In summary, the submodel couplings are: M3 to M4 (options feed election setup); M4 to M6 (the election configuration feeds aggregation); M7 to M2 (shocks are absorbed into the preference updates); M2 to M5 (evolved preferences feed vote-casting); M1 to M6 (offline weights and/or goal feed aggregation); M5 to M6 (ballots feed aggregation); and M6 back to M2 (results feed in as Events to the next preference loop).

3.2 Pre-Election Stage

The pre-election stage runs from the end of the previous election cycle (or simulation initialisation) until the election is opened. It encompasses the formal election setup and the per-timestep preference and belief dynamics (M2) that precede a voting event. Its function is to configure the election, generate the options voters will vote on, and evolve voter preferences (and, in the Active Inference architecture, beliefs) to the state they will be in when voters vote.

- **M3 (Option Generation)** executes at campaign initiation, constructing the set of Proposals from empirical candidate data (Aarhus pilot). Proposals are passed to M4.
- **M4 (Election Setup)** configures the Election entity: it specifies the voting method and electoral system (set independently) and the vote decider, the ballot structure, the number of stages, and the set of alternatives drawn from M3. In experiments testing institutional design, the same preference landscape and proposal set can be passed through multiple M4 configurations, enabling direct comparison of electoral system properties.
- **M2 (Preference and Belief Dynamics)** runs iteratively across the pre-election period. At each timestep, each Voter executes perceive() → learn() → updatePreferences() in response to incoming stimuli: event broadcasts from the Polis, peer communications across the social network, and campaign information. This is the primary mechanism by which preferences evolve toward the election.
- **M7 (Exogenous Change),** when active, fires at scheduled or stochastic points during this stage. Shocks are absorbed through the standard M2 update cycle; no separate processing pathway is required.
- **M1 (Quantum Voting)** is an offline quantum-theoretic component that supplies the two-boundary form of the aggregation operator and can exchange influence weights and/or a goal-constraint with the classical model. Its exact role and direction are configurable.

3.3 Election Stage

The election stage is the act of individual vote-casting. It is bounded by the opening and closing of the ballot.

- **M5 (Voting)** executes once per voter. Each Voter retrieves its current preference state (as delivered by M2), runs forecast() to estimate the likely election outcome, then executes collapsePreferences() to collapse its probabilistic preferences into a discrete vote (the forward boundary). Where naive Theory of Mind is active, voters additionally down-weight candidates with low forecast viability, producing strategic voting behaviour. The institutional/collective goal ⟨φ| is not applied at this per-voter step; it is applied at aggregation (M6), so M5 emits a sincere or strategically adjusted forward vote only. Each voter submits its vote to the Election entity.

3.4 Post-Election Stage

The post-election stage covers vote aggregation, outcome evaluation, and the feedback of results into voter beliefs. Its outputs are both the primary observable quantities for validation and the stimuli that drive the next pre-election cycle.

- **M6 (Election)** aggregates submitted votes using the mechanism configured by M4 and applies the two-boundary operator: the aggregated forward preferences (p_fwd) are combined with the institutional goal ⟨φ| (p_bwd = Election.backwardGoal) and renormalised, p = p_fwd · p_bwd / Z, classically. A flat backwardGoal reproduces goal-free aggregation. M1, an offline quantum-theoretic component, supplies the two-boundary form of this operator and may exchange the forward weights and/or the goal-constraint ⟨φ| with the classical model, with the exact role and direction configurable. M6 then computes the outcome evaluation metrics: PerceivedFairnessScore, TrustIndex, representation quality, and divergence from empirical baselines where applicable.

- **M6 broadcasts results** to all voters as an Event via the Polis. This is the coupling point that closes the loop: the broadcast re-enters the pre-election stage of the next cycle, where voters perceive the outcome, compare it to their prior forecast, and update their generative models via M2. The magnitude of this belief revision, governed by the discrepancy between expected and realised outcomes, is a key variable in the simulation's account of how electoral experience shapes political behaviour over time.

4. Design Concepts

4.1 Basic Principles

The framework is grounded in three theoretical traditions: (i) Social Choice Theory, providing normative criteria for evaluating voting mechanisms (Arrow's conditions, fairness axioms, Condorcet efficiency); (ii) Active Inference and the Free Energy Principle, providing a computationally grounded account of individual belief formation and decision-making; and (iii) quantum information theory, contributing several distinct and independently usable resources (richer preference encoding via superposition, correlation and strategic voting via entanglement, genuine randomness, optional security and privacy, and a natural outcome-side boundary condition) that may inform or substantiate classical agent-based modelling. The framework draws on these modularly rather than depending on any single one; they are detailed in §7.1.

A fundamental design decision is that the quantum component operates exclusively offline: it supplies the two-boundary form of the aggregation operator (which originates in the ABL two-boundary rule, a pre/post-selection structure) and, where configured, exchanges a goal-constraint and/or forward influence weights with the classical model as classical parameters. Quantum entanglement is a separate, distinct resource, used where applicable to represent correlations between voters' preferences that do not factor into independent classical probabilities. No live quantum state manipulation occurs at runtime. This is motivated by three considerations: (i) current quantum hardware cannot execute the required computations at electoral scale; (ii) the operator is the classical projection of the two-boundary rule, so the aggregation runs classically and the research question concerns the goal-conditioned form and the chosen weight source, not whether quantum computation per se changes behaviour; (iii) the offline architecture keeps full compatibility with classical simulation infrastructure and lets different weight sources (earned-influence, quantum-derived, uniform) be compared in controlled experiments.

4.2 Emergence

The primary emergent phenomena of interest are: (i) macro-level preference distributions and polarisation from micro-level belief updating and social influence; (ii) collective electoral outcomes— including strategic voting patterns and median voter deviations— from individual forward votes aggregated by the two-boundary operator in M6; (iii) trust and legitimacy scores from the aggregation of individual utility assessments of election results; and (iv) coalition formation and group-level consensus dynamics from voter communication patterns. None of these are directly imposed; they arise from the interaction of voter-level processes and institutional structures.

4.3 Adaptation

Voters adapt their beliefs and preferences in response to environmental stimuli via the learn() and updatePreferences() methods. In M2's Active Inference architecture, this takes the form of minimising variational free energy through predictive coding: voters update their generative model to reduce the discrepancy between predicted and observed sensory states. Voters also exercise strategic

adaptation in M5 (Voting) via naive Theory of Mind: they observe simulated polling information and down-weight candidates with low perceived electoral viability, producing classic strategic voting behaviour.

4.4 Objectives

Voters do not have explicit utility functions in the classical economic sense. In M2's Active Inference architecture, behaviour is governed by the minimisation of expected free energy under a fixed prior preference, which encodes what the voter 'wants' the world to look like. This prior is the voter's own desired outcome and belongs to the forward boundary it expresses (via M2); it is distinct from the collective backward boundary ⟨φ|, which is the institutional goal applied at aggregation (M6). Voting decisions are probabilistic, driven by how much a candidate is expected to reduce the discrepancy between the voter's desired and believed world-states. In classical Social Choice submodels, voters are modelled as expressing sincere preferences over alternatives, with optional strategic overlays.

4.5 Learning

Learning operates at two timescales. Within an election cycle, voters update their preferences (and, in the Active Inference architecture, beliefs) in response to events and peer communications (M2 per-timestep loop). Across election cycles, voters integrate election outcomes as Events, revising their internal state based on the difference between expected and realised outcomes. In the Active Inference architecture, the generative model structure (whether Markov Decision Process, partially observable MDP, or hierarchical) is defined per experiment and constitutes a major research variable; this choice is consequential for the learning under investigation: in a fully observable MDP the true state is always known, so voters never misjudge the situation they are in, and the cross-cycle self-correction of interest arises only under partially observable or hierarchical structures.

4.6 Prediction

Voters execute forecast() to predict the likely outcome of an election before voting. This forecast informs the voter's strategic (Theory-of-Mind) vote adjustment, which enters the forward boundary that the two-boundary operator then combines with the institutional/collective goal at aggregation (M6). The accuracy of voters' predictive models is itself a research variable: experiments will compare outcomes under perfect information, poll-based information, and information-deprived conditions.

4.7 Sensing

Voters perceive: (i) event broadcasts from the Polis (exogenous shocks, election results); (ii) communications from network neighbours (peer influence, information diffusion); (iii) polling information (aggregated preference signals, in experiments where this is active). Voters do not directly observe the preferences of other voters; all social information is mediated through communication and broadcast mechanisms. The scope of sensing is parameterised by the voter's connectedTo network and the targetGroup of each Event.

4.8 Interaction

Voter–voter interaction is network-mediated. Voters communicate preferences and information through initiateComms() (bidirectional, targeted) and broadcast() (unidirectional, group-level). Group-level gravitas modulates the influence weight of broadcast messages. Network topology is a configurable parameter; supported structures include Erdős–Rényi random graphs, Watts–Strogatz

small-world networks, and Barabási–Albert preferential attachment networks, with configurable homophily parameters.

4.9 Stochasticity

Stochasticity is present at multiple levels: (i) random seeding of initial preference profiles in abstract scenarios; (ii) probabilistic per-voter preference collapse via collapsePreferences() (the classical softmax); (iii) stochastic event timing and impact in M7; (iv) communication network formation. In empirical scenarios, stochasticity is reduced by calibrating initial conditions from survey and census data. All random processes are seeded for reproducibility. Where the quantum component (M1) is engaged, it introduces a formally distinct potential source of indeterminacy in the framework, whose implications for simulation outcomes are a research question.

4.10 Collectives

Groups are explicit first-class entities, not emergent aggregations. They serve multiple roles: demographic strata (for census-calibrated initialisation), ideological blocs (for preference clustering), political parties (as Proposal generators in M3), and media entities (as broadcasters of targeted information). Group membership is not fixed: voters may update their group affiliations in response to preference changes, though the specific rules for this are defined per experiment.

4.11 Observation

The following outputs are logged at each timestep and/or at each election event:

- Voter-level: preference vectors, belief states, vote decisions, group affiliations.
- Group-level: aggregate preference distributions, polarisation indices, participation rates.
- System-level: election outcomes (seat/vote shares), PerceivedFairnessScore, TrustIndex, divergence between simulated and empirical outcomes (in validation scenarios).

Output handling: configurable CSV exports; event-query reports; data pipeline to visualisation tools including the Simulation Player for replay of simulation scenarios.

5. Initialisation

5.1 Abstract Scenarios

For theoretical parameter exploration, voters are initialised with randomly sampled or systematically varied preference vectors drawn from specified distributions (e.g., uniform, multimodal, polarised). Network topologies are generated algorithmically. The number of voters, number of policy dimensions, and distribution parameters are sweep variables. No empirical data are required.

5.2 Empirical Scenarios (Aarhus Pilot)

For realism-oriented scenarios, initialisation draws on three empirical sources collected in the Aarhus pilot:

- Candidate preferences: sourced from the Kandidattest (Municipal election candidate questionnaire), providing policy positions across standard Danish political dimensions for actual candidates.

- Voter preferences: collected in a survey conducted in March 2026, providing preference profiles for a sample of Aarhus voters.
- Media reports: used to seed event schedules and to calibrate the content and magnitude of information shocks.
- Additional Experiments developed by Aarhus Kommune in late 2026.

Synthetic voter populations (WP5) are generated to represent the full Aarhus electorate (~180,000 voters) calibrated to Statistics Denmark (Danmarks Statistik) census distributions on age, education, employment, and geographic distribution. LLM-generated preference profiles are used to extend the empirical seed data to the full synthetic population.

5.3 Quantum Module Initialisation

In experiments that engage M1, the offline quantum-theoretic component runs before simulation execution and exchanges its outputs with the classical model at the pre-election pause point. Candidate exchanges include the two-boundary operator form, a goal-constraint, and a forward influence-weight vector {w_i}; the exact role, direction, and inputs/outputs are configurable.

6. Input Data

Data Type	Source	Use
Candidate preferences	Aarhus Municipal Elections Kandidattest; candidate-test responses from parliamentary candidates in Aarhus collected from Altinget, DR.dk, and TV2	Seed candidate-voter preference profiles (empirical scenarios); Proposal attributes (M3)
Voter preferences	Aarhus voter survey, March 2026	Calibrate initial voter preference distributions
Demographic structure	Statistics Denmark (Danmarks Statistik); Aarhus Open Data	Stratified voter population initialisation
Electoral results	Danish Election Database (Valgdatabase.dst.dk);	Validation of simulated outcomes
Network topology priors	Literature (empirically informed; small-world, scale-free)	Voter network initialisation
Voting system specifications	Literature review (D4.1 internal references)	M4 election configuration (ballot structures, aggregation rules)
Voter Behaviour under different voting systems	Embedded voting experiment in Aarhus voter survey, March 2026; planned lab and field experiments with Aarhus municipality (WP8)	Empirical priors and initial behavioural inputs for how voters express preferences under different voting mechanisms; informs the M4 - M6 pipeline.
Synthetic voter profiles	WP5 synthetic data pipeline (LLM-generated)	Population scale-up; theoretical scenario seeding
Quantum module parameters	M1 offline output (pre-simulation), where engaged	Configurable operator inputs (form, goal-constraint, and/or influence weights) in quantum experiments

Table 10. *Input data sources.*

The Aarhus pilot data collection (candidate preferences from Kandidattest, voter survey March 2026, media reports) grounds the simulation in a specific, well-documented political context while preserving the generality of the framework for theoretical analysis. The Danish electoral context is particularly suited for this purpose: high data availability (Statistics Denmark, The Danish Election Database), a relatively transparent political culture, and the Aarhus City Council's involvement as a project partner enabling access to participatory governance experiments.

The Aarhus voter survey was fielded in March 2026 through Norstat and targeted eligible voters residing in Aarhus Municipality. The survey was launched immediately after the national general election, with the first response collected only one day after election day. It was designed as the main voter-side empirical input for the models. The survey collected respondents' positions on national and local political issues using a candidate-test-style issue battery, allowing voter preferences to be mapped onto the same issue space as candidates and parties. It also measured general attitudes toward the political system, voting experience, turnout and vote choice, and demographic characteristics. These data provide the basis for estimating the distribution of voter preferences in Aarhus and for calibrating voter attributes and initial preference profiles in simulation scenarios.

Candidate-side data were collected separately from Danish candidate-test platforms. In addition to the municipal Kandidattest data, candidate-test responses from candidates running for the national parliament in Aarhus were collected from three major Danish platforms: Altinget, DR.dk, and TV 2. These data provide empirical information on candidate and party positions across comparable issue dimensions, allowing the model to define proposal and candidate attributes more precisely.

In addition to the main preference and attitude measures, the Aarhus data collection includes experimental components designed to examine how citizens respond to different voting mechanisms. The March 2026 voter survey included an embedded voting experiment comparing how respondents would vote under the current Danish proportional representation system and under a quadratic voting alternative. This will be complemented by planned lab and field experiments developed in collaboration with Aarhus Municipality as part of WP8. These experiments are designed to examine how people's expression of preferences varies across alternative voting rules. Together, these data will provide empirical priors and initial behavioural inputs for the M4-M6 pipeline, where voter behaviour under different voting rules needs to be represented.

7. Submodels

This section provides detailed descriptions of each of the seven submodels constituting the EuropeanCity2 constellation. Each submodel can be developed, tested, and published independently. Integration across submodels occurs through defined I/O interfaces (CSV exchange at pause points; shared entity state). The coupling architecture is described in §3.3 and §7.8.

7.1 Model 1: Quantum Voting

Purpose

Model 1 is an offline quantum-theoretic component of the framework. In the simulation it functions as an offline solver: it supplies the two-boundary form of the aggregation operator (and of the associated weighting, where applicable) that the classical model uses and, where configured, exchanges a goal-constraint and/or forward influence weights with the classical model in either direction, in each case as classical parameters. It is not a runtime quantum computer, and its exact role and direction are configurable. Quantum information theory offers several further resources that an experiment may

configure M1 to explore (richer preference encoding via superposition, entanglement for voter correlations, genuine randomness, and optional security and privacy); the framework draws on these modularly, not all of them depend on the two-boundary (TSVF) structure, and in the offline architecture they enter the classical model as derived parameters or comparison conditions rather than runtime quantum machinery (see Research Variables).

Process

The core mathematical object is the two-boundary operator p_fwd · p_bwd / Z, where p_fwd is the forward (expressed preference) distribution, p_bwd is the backward boundary ⟨φ| (the collective goal over outcomes), and Z is a normalisation constant. The operator is evaluated classically at aggregation (M6, §7.6), not per voter. The backward boundary ⟨φ| encodes the collective fairness goal: what the election outcome should look like if it is to satisfy the specified normative criteria.

This two-boundary form is one of the contributions M1 supplies to the classical model (see the Purpose and Research Variables). Quantum encoding can also structure preferences into superposition states and use entanglement to represent correlations between voters. Beyond supplying the form, M1 may exchange the goal-constraint ⟨φ| and/or the forward influence weights {w_i} with the classical model; the exact role and direction are configurable. M1 operates offline, and no live quantum computation occurs at runtime.

Inputs/Outputs

Configurable, depending on the role assigned to M1: candidate exchanges include the two-boundary operator form, a goal-constraint ⟨φ|, and a forward influence-weight vector {w_i}, passed between M1 and the classical model in either direction.

Integration

M1 connects to the classical model offline. It supplies the two-boundary operator form and, where configured, exchanges a goal-constraint ⟨φ| and/or an influence-weight vector with M6, which applies them in the aggregation operator. The exact exchange is configurable. M1 is run as a pre-simulation solver; no live quantum computation occurs at runtime, and the interface is a structured CSV exchange at the pre-election pause point.

Research Variables

The distinct quantum resources the framework can be configured to explore, drawn on modularly: (1) preference encoding as superposition states, richer than classical alternatives and the basis of the quantum-analogue Arrow bypass demonstrated in the consortium's quantum-voting work; (2) entanglement structure, representing voter correlations and strategically coordinated voting that do not factor into independent classical probabilities; (3) quantum measurement as a source of genuine, unbiased randomness (for example tie-breaking or sampling); (4) optional quantum protocols for ballot security and privacy; and (5) the two-boundary (ABL/TSVF) outcome-side boundary condition that supplies the operator form applied at aggregation, which is the classical projection of the two-boundary rule and reproducible without quantum hardware (see §4.1), so the framework does not depend on it being quantum. None is required, and in the offline architecture each enters as a derived parameter or a comparison condition. Further variables: fairness-criterion specification; comparison of weight-assignment sources (earned-influence, quantum-derived, uniform, utilitarian); quantum vs. classical quadratic voting outcomes.

7.2 Model 2: Preference and Belief Dynamics

Purpose

Model 2 implements the per-timestep cognitive process by which voters form and update their preferences (and, in architectures that use one, their beliefs). It is the micro-level foundation for all downstream voting behaviour and the primary model for polarisation dynamics, social influence, and the effect of exogenous shocks. M2 is a shared interface with two selectable architectures: data-driven cognitive coherence (Architecture A) and Active Inference (Architecture B). The choice of architecture is itself a research variable.

Process

Regardless of architecture, M2 runs a per-timestep loop on each voter built from three operations:

- perceive(): receives stimuli from the environment (event broadcasts, peer communications) and the voter's prior preference state (and, where the architecture uses one, a belief state).
- learn() / updatePreferences(): revises the voter's preferences (and belief state, where used) in response to the incoming stimuli.
- broadcast() / initiateComms(): transmits information to network neighbours, who receive it as stimuli in their own perceive() step.

The within-timestep ordering of these operations is architecture- and implementation-specific. Under the engine's one-tick rule (a message broadcast at step t is perceived by others at step t+1) either ordering is permitted, but the choice is not inert: broadcasting before the update emits the voter's pre-update state, broadcasting after emits its post-update state. Architecture A broadcasts before it perceives and updates; the Active Inference architecture perceives and updates before it acts.

Selectable Architectures

Architecture A, data-driven cognitive coherence (Lorenz): preferences over the named policy topics are updated toward coherence under a fixed topic-correlation matrix (a "mental model"), driven by simple single-topic messages and calibrated from the voter survey. There is no separate belief layer: the model is preferences-only, which matches the empirical municipal data.

Each Voter holds a preference vector x over the n_pref named policy topics (continuous values in the range [−2, 2]; the reference implementation uses the ALCL topic set; n_pref is configurable). The Voter also carries a static mental model C, a symmetric topic-correlation matrix with zero diagonal, initialised from empirical survey correlation matrices stratified by demographic subgroup (see Section 5.2). The coherence of a preference state is the quadratic form c(x) = x^T C x.

The unit passed between Voters is a message: a single topic-value pair {topic i, value v}. The Polis broadcasts one message per timestep (campaign or news input; see Section 6), and each Voter broadcasts one message per timestep (optionally prioritising its topics of interest). On receiving a message about topic i with value v, updatePreferences() forms the candidate state x_hat (equal to x with component i set to v), computes the coherence change Δc = c(x_hat) − c(x), accepts the move with logistic probability P = 1 / (1 + e^(−k·Δc)), and on acceptance updates that component by the convex combination x_i := (1 − α)·x_i + α·v. Coherence-increasing messages are thus accepted with probability above one half and coherence-decreasing ones below it.

Architecture B, Active Inference / free energy (Legrand): each voter carries a generative model with beliefs over hidden world-states; learn() minimises variational free energy to update those beliefs, and updatePreferences() revises the preference distribution, with preferences encoded as the prior over observations. The generative model structure (a Markov Decision Process, a partially observable MDP, or a hierarchical model) is itself a variable.

The two architectures are compared on their ability to reproduce empirically observed preference dynamics (polarisation trajectories, volatility patterns).

Inputs/Outputs

- Input: event broadcasts; peer communications; prior preference state (and belief state, in Architecture B).
- Output: updated preference distribution (consumed by M5 at election time); updated belief state where Architecture B is used.

Integration

M2 is triggered at every timestep and after every election event (when M6 broadcasts election outcomes). It is the primary consumer of M7 (Exogenous Change) outputs and a key supplier to M5 (Voting). It operates independently in Year 1 and is integrated with M5 in Year 2.

Research Variables

Choice of M2 architecture (cognitive coherence vs Active Inference); for Architecture A, the topic-correlation (mental-model) structure and message model; for Architecture B, the generative model structure and free-energy algorithm; homophily parameters; network topology; cross-validation against empirical polarisation data.

7.3 Model 3: Option Generation

Purpose

Model 3 generates the set of Proposals (candidates, policies, ballot measures) that voters will evaluate and vote on. In the context of Danish elections, this is primarily the process by which political parties nominate candidates and establish policy positions.

Process

In Year 1 and Year 2, M3 is data-driven: Proposals are constructed directly from the empirical Aarhus pilot dataset (DR Kandidattest entries and candidate policy positions from actual election ballots). Proposal attributes (policy dimension values) are populated from this data.

In later experimental phases, LLM-based endogenous option generation will be explored. In this mode, an LLM is prompted with contextual information (current aggregate preference distributions, group demands, political context) to generate plausible candidate or policy profiles. This enables counterfactual scenario analysis: what if different candidates had run, or parties had adopted different platforms?

In deliberative scenarios (citizen assemblies, participatory budgeting), M3 models the proposal generation process itself: groups of agents collectively formulate and refine proposals through iterative communication rounds, with sortition or other selection mechanisms determining which agents participate.

Inputs/Outputs

- Input: Empirical candidate data (Aarhus pilot); LLM prompts with current preference distributions (in LLM-extended mode); agent group structure (for deliberative scenarios).
- Output: Populated Proposal objects with attribute vectors; ballot structure for M4.

Integration

M3 outputs constitute the alternatives input for M4 (Election Setup). It is activated at campaign initiation, prior to the election sequence. In Year 3, M3 may receive feedback from M6 election outcomes to model party adaptation of platform positions across election cycles.

Research Variables

Data-driven vs. LLM-generated proposals; diversity of option space; impact of sortition vs. representative deliberation on proposal quality; sensitivity of electoral outcomes to option generation mechanism.

7.4 Model 4: Election Setup

Purpose

Model 4 handles the technical configuration of an election: it specifies the voting system, ballot structure, timing, and protocol. It does not model cognitive or preference processes but serves as the institutional design layer, enabling systematic comparison of different electoral system architectures.

Process

Given a set of Proposals (from M3) and a specified voting method, electoral system, and vote decider, M4 configures the Election entity:

- Selects the voting method (the ballot mechanic): single-choice or quadratic voting in v1; ranked-choice, approval, and others in later phases.
- Selects the electoral system (the aggregation rule): open-list proportional representation or plurality in v1. The voting method and electoral system are independent axes and combine freely (for example single-choice with plurality is first-past-the-post; quadratic with proportional is also valid).
- Selects the vote decider (the swappable per-voter rule that converts each voter's per-voter probabilistic result into a ballot or abstention): keep-sampled-vote (default) or threshold-abstain.
- Defines the ballot structure: number of choices per voter, ranking depth, weight allocation rules (for QV/QQV).
- Sets the electoral timeline: number of stages (primary, run-off), campaign duration, information availability windows.
- Issues ballots to voters: instantiates the alternatives list available to each voter.

Inputs/Outputs

- Input: Proposal set (from M3); voting-method and electoral-system specification (from experiment design); quantum weight vector (from M1, in QV experiments).
- Output: Configured Election entity; voter ballot assignments.

Integration

M4 is activated once per election event, prior to M5. It is the primary locus for institutional design stress-testing experiments: the same preference landscape (from M2) and proposal set (from M3) can be run through multiple M4 configurations to compare institutional outcomes.

Research Variables

Voting mechanism and electoral system; number of stages; information availability; ballot complexity; comparison of QV and QQV weight allocation under varying preference distributions.

7.5 Model 5: Voting

Purpose

Model 5 implements the micro-level cognitive process of individual vote casting. It is the primary locus of Active Inference in the voting act itself, and the integration point between the cognitive model (M2), the quantum weight assignment (M1), and the electoral mechanism (M4/M6).

Process

With the election open (configured by M4), each voter:

- Retrieves its current preference distribution from its internal state (as updated by M2).
- Retrieves the election configuration and ballot structure from M4. This defines the action space of voters: for instance, whether an agent casts a single vote, options available on ballot, decides response strategies given the ballot structure, and casts the ballot based on the behaviour function.
- Executes forecast(): predicts the likely election outcome using available information (own preferences, polling data if available, network signals). This produces a probability distribution over candidate/proposal outcomes.
- Executes collapsePreferences(): collapses the voter's preference distribution into a discrete vote on the forward boundary (sincere, or strategically adjusted by the Theory-of-Mind step below). The softmax function converts the preference scores into a voting probability, from which the vote is sampled. The institutional/collective goal ⟨φ| is not applied at this per-voter step; it enters at aggregation (M6).
- Applies naive Theory of Mind (optional): down-weights candidates with low forecast probability of winning, implementing strategic voting behaviour.
- Executes vote(): submits the discrete vote to the Election entity.

Each voter's fixed preferences encode what they want the world to look like (prior over desired states). The dissatisfaction score for a candidate is the expected divergence between the voter's desired world and the world the voter believes would result from the candidate being elected. This score is the basis for the voting probability.

Inputs/Outputs

- Input: Voter preference distribution (from M2 state); election configuration and ballot structure (from M4); voter-response priors from the Aarhus voter survey, embedded voting experiment, and WP8 experiments; quantum weights (from M1, if active); polling information (if available).
- Output: rule-specific ballot action per voter, submitted to Election entity.

Integration

M5 is triggered once per election. In Year 2, the preference state input is provided by the integrated M2 generative model. In Year 3, M1-derived forward weights enter as indGravitas, the goal ⟨φ| being applied separately at aggregation in M6, and the strategic component may be extended with more sophisticated social inference.

Research Variables

Generative model architecture for belief state; presence/absence of naive Theory of Mind; information availability; quantum vs. classical preference collapse; vote choices per electoral rules, calibration against empirical voting data from Aarhus pilot survey and WP8 experiments.

7.6 Model 6: Election

Purpose

Model 6 aggregates individual votes (from M5) into collective outcomes using configurable electoral systems, evaluates the results against normative and empirical criteria, and broadcasts outcomes back to voters to initiate the next belief-updating cycle.

Process

M6 executes through three sequential phases:

Phase 1 — Vote Aggregation: The Election entity's calculateResults() method aggregates submitted votes using the configured mechanism (plurality, ranked-choice/instant-runoff, proportional representation [D'Hondt, Sainte-Laguë], quadratic voting, or quantum-inspired quadratic voting). This is the step at which the two-boundary operator is applied: the aggregated forward preference distribution over outcomes (p_fwd) is multiplied by the institutional/collective goal (p_bwd = Election.backwardGoal) and renormalised, $p = p_{fwd} \cdot p_{bwd} / Z$, yielding the goal-conditioned collective outcome. A flat/uniform backwardGoal reproduces ordinary goal-free aggregation as the special case. The operator is evaluated entirely classically. M1, where engaged, supplies the two-boundary form and may exchange a goal-constraint $\langle\varphi|$ and/or forward influence weights as offline-derived parameters. The forward-side influence weights (indGravitas) enter p_fwd and are distinct from the backward goal applied here; their source is configurable (earned in-simulation, M1-supplied, or flat).

Phase 2 — Outcome Evaluation: The Polis computes:

- calculateFairnessScore(): PerceivedFairnessScore—voter-level perceptions of the election's fairness, aggregated to a system mean.
- calculateAggregatedScores(): TrustIndex, polarisation post-election, representation quality (divergence between population preferences and winning outcome), and participation rate.
- Divergence metrics: comparison of simulated outcomes against empirical baseline (in validation scenarios).

Phase 3 — Results Broadcast: The Polis broadcasts election outcomes as an Event to all voters. This triggers M2 (Belief Dynamics): voters perceive the outcome, compare it to their forecast, and update their generative models accordingly. This feedback loop is the mechanism by which elections influence subsequent preference formation and political behaviour.

Inputs/Outputs

- Input: Vote submissions from all voters (M5 outputs); electoral mechanism configuration (M4); forward influence weights and a goal-constraint (source configurable; M1-supplied where engaged).
- Output: Election result (seat/vote shares, winning proposals); fairness and trust metrics; outcome broadcast to voters.

Integration

M6 is the terminal step of the election sub-process and the initiator of the post-election belief update cycle. It provides the primary observable outputs for validation against empirical electoral data. It also serves as the testing ground for integration with M1 in Year 3.

Research Variables

Electoral mechanism type; quantum vs. classical aggregation; fairness metric specification; sensitivity of outcomes to voter weight distributions; median voter theorem compliance under varying preference landscapes.

7.7 Model 7: Exogenous Change

Purpose

Model 7 introduces dynamism into the simulation by modelling the impact of unexpected external events—political shocks, scandals, policy announcements, economic crises—on voter beliefs and preferences, and thus on subsequent electoral behaviour.

Process

An exogenous shock is represented as an Event with a specified impact vector and targetGroup. When fired, M7 executes the standard M2 preference/belief update cycle (perceive → learn → updatePreferences) for the affected voters, with the shock's impact vector as the observed stimulus. The forward preference state moves in response to the shock, while the backward boundary ⟨φ|—the collective normative goal—remains fixed, anchoring the next election's evaluation criteria to the pre-shock ideal.

Shock magnitude, direction, and targeting are experimental variables. Targeted shocks (affecting specific demographic groups) enable modelling of differential political mobilisation. System-wide shocks enable stress-testing of institutional resilience. The decay rate of shock effects over subsequent timesteps is a key parameter.

Inputs/Outputs

- Input: Shock specification (magnitude, direction, targetGroup, date); current voter belief/preference states.
- Output: Updated voter belief/preference distributions; these feed directly into M5 and M6 in subsequent election cycles.

Integration

M7 is triggered by scheduled or stochastic events in the Polis event schedule. It feeds into M2's standard update cycle; there is no separate processing pathway. In Year 3 full integration, M7 shocks can be chained: an election outcome (M6) can itself constitute a shock that triggers M7, creating endogenous political feedback dynamics.

Research Variables

Shock magnitude and decay; targeting specificity (universal vs. group-specific); endogenous vs. exogenous shock generation; tipping point analysis for democratic transitions; comparison with empirical shock events from the Aarhus pilot.

8. Model Evaluation

8.1 Validation Strategy

Validation is acknowledged as a particular challenge for this framework, given its theoretical ambition and the novelty of the quantum and Active Inference components. The validation strategy operates at three levels:

Submodel validation

Each submodel is validated independently against the patterns identified in §1.3 before integration. M2 is validated against empirical preference distribution and vote choice data from the voter survey. M5 and M6 are validated against Danish electoral outcomes (vote shares, participation rates). M1 is not validated against empirical data: its contribution is the mathematical form of the two-boundary operator, so it is checked by verifying that the derivation is correct and reduces to the expected special cases (for example, ordinary voting under a flat goal). Where M1 is configured to produce a weight vector or a goal, those outputs are checked by confirming they satisfy the fairness or goal criteria they were computed to meet.

For the voting-related components, submodel validation also includes confirming that the electoral systems configured in M4 and implemented through M5–M6 produce expected outcomes under empirical and theoretical benchmark contexts, including observed preference distributions and theoretically-validated preference profiles. The embedded voting experiment in the Aarhus voter survey and the planned WP8 experiments provide empirical priors and potential validation benchmarks for how voters respond to different voting mechanisms. The WP8 experiments, for instance, are designed to examine similar questions across lab and field settings, which provides opportunities to compare whether observed patterns of voting behaviour are robust across experimental contexts.

Integration validation

Integrated model constellations are validated against the full set of empirical patterns from the Aarhus pilot. Sensitivity analysis identifies which parameters drive divergence from empirical patterns.

Predictive validation

Where feasible, the simulation will be used to generate a priori predictions for future governance processes (e.g., the next Aarhus City Council election or a participatory budgeting process), with post-hoc comparison to actual outcomes.

8.2 Calibration

Parameter calibration proceeds from coarse to fine: demographic parameters are calibrated directly from census data; preference distributions are calibrated from the voter survey; network topology parameters are calibrated from literature priors (small-world coefficients for political communication networks). Where M1 is engaged, its outputs are derived analytically from the specified input criteria rather than calibrated empirically from real world data.

8.3 Uncertainty and Sensitivity Analysis

Given the scale of the parameter space, uncertainty analysis will prioritise: (i) parameters with high theoretical uncertainty (Active Inference generative model architecture, quantum entanglement structure); (ii) parameters to which key outcome metrics (polarisation, trust, vote shares) are most sensitive, identified via variance-based sensitivity analysis (Sobol indices); and (iii) structural uncertainties in model integration (which submodels to couple, how tightly).

9. Planned Extension

This section describes how voluntary-contribution-based indGravitas would plug into the existing D6.3 seams without changing the v1 architecture. It is collected here as one module so its couplings are visible before being folded into §§2–8. It adds one capability: the per-Voter weight indGravitas (Table 2) can be set from a Voter's voluntary contribution accumulated in-simulation. This is one configurable weight source, not a change to the framework; §7.6 already calls the source configurable. The framework stays neutral over sources: flat (weight 1; one-person-one-vote; the default baseline), earned (this module), M1-supplied (§7.1), and self-benefit (inverted control). The earned setting departs from equal suffrage and is specified as a testable comparand judged by measures it does not define (§8), not a recommendation; the experiments live in the accompanying study, not here.

Entities and state variables (§2). Voter (Table 2): contribution, wealth (contribution is measured against total wealth, not income), AVTR (= mean contribution/wealth), t (periods sustained), giving indGravitas = 1 + AVTR*t; and contributionPropensity and judgment as two independent traits, so

their correlation is a settable parameter, not a built-in identity. Polis (Table 7), new fiscal layer: cityBudget, taxBase, obligatoryTaxRate, publicGoodsProvision.

Actions and events (§3, §7). contribute(): a Voter action via selectAction(), its size produced by the active M2 architecture (§7.2), a choice the Voter makes rather than a value assigned. Earned-influence-law Event (Table 6): when fired, switches the Election from flat to earned weighting by reading AVTR*t into indGravitas (§7.7).

Process (§3, §7.6). indGravitas enters the M6 combine step as a forward-side weight in p_fwd, distinct from ⟨φ|; it is a single weight object, optionally exported to M1 as gravitas, not a second weight. The update schedule, normalization, and persistence of indGravitas are implementation details, deferred to the D6.3 build and not fixed by this protocol. New Polis computation each period: obligatoryTaxRate = (cityBudget − Σ contributions) / taxBase, so contributions lower the obligatory levy mechanically (budget held fixed; only contributions vary). The M6 broadcast (§3.4) carries the period's obligatoryTaxRate and publicGoodsProvision change via the Event impact vector, so Voters perceive() and attribute the fiscal consequence.

Design concepts (§4). Evaluated under the partially-observed/hierarchical generative model already named in §4.5 (the structure required for cross-cycle self-correction); foregrounds an existing option, no new machinery.

Initialisation and input (§5, §6). The Aarhus participatory-budgeting episode (§1.1, §8.1) pins cityBudget with the real pot and adds a voluntary-augmentation option.

D6.3 v1 seams. Voter variables → cognitive-state / per-voter attribute extensions; indGravitas in p_fwd → per-ballot influence weighting; the law Event → exogenous-stimulus seam; the fiscal layer → extend-the-environment / aggregations seam; broadcast re-entry → post-election-feedback seam (metrics on AbstractElectionResultAction); layered model → belief-dynamics seam and reserved mental-model slot; participatory budgeting → the named PR/budget electoral seam. contribute() is added as a new action without modifying any existing code.