Corralling AI Risks with Grassroots Social Contracts (Full Version)

Ehud Shapiro
London School of Economics and Weizmann Institute of Science
UK and Israel

Abstract

AI's inflection point is a fountain of alarming risks. Digital platforms in particular are vulnerable to three major AI risks, as those in control of global platforms cede control to AI, of the algorithms, of membership and of content:

(1) **AI control of global platforms**—social and economic—used by the majority of humanity, resulting in the majority of humanity becoming vulnerable to AI control.
(2) **Unaccountable AI agents** operating anonymously on social and economic digital platforms, side-by-side with humans, possibly masquerading as humans, serving unknown masters with unknown goals, operating under unknown restraints without accountability or culpability.
(3) **Unattributed deepfakes** generated by anonymous AI, with people unable to distinguish them from genuine content and with their producers evading accountability and culpability.

Here, we propose to corral these AI risks with grassroots social contracts—a digital incarnation of a two-centuries old notion of Pierre-Joseph Proudhon. Proudhon envisioned a social contract that can be freely-adopted by any group of people, where groups bound by the same social contract can union by mutual consent, possibly into a single global union. Grassroots social contracts are realised by grassroots platforms—smartphone-based person-to-person digital platforms owned, operated and governed by their participants. Fundamental grassroots platforms include the grassroots social graph, grassroots social networks, child-safe social networks, and grassroots currencies, with higher-level platforms providing for democratic digital communities and "sharing-economy" cooperatives. We developed, with AI, a methodology to compile grassroots social contracts to working smartphone apps. We present grassroots social contracts for a social graph with accountable AI agents and for a social network with digital provenance, and working smartphone apps that implement the two and show how they corral these major AI risks:

(1) **AI control of global platforms** is mitigated by grassroots platforms by design, as, unlike global platforms, they are not controlled by a third party who may then cede control to AI.
(2) **Unaccountable AI agents** are excluded, as AI agents are introduced to the grassroots social graph with accountability: similarly to cars, their manufacturers are known and are accountable for manufacturing defects, and their operators are known and are accountable for reckless operation.
(3) **Unattributed deepfakes** cannot circulate in grassroots social networks: every item carries the signatures of its author and of every agent that forwarded it, so creators and forwarders of harmful content are accountable and culpable.

Grassroots platforms are presently a concept with mathematical and computational theory, working prototypes, and tools for their large-scale simulation and testing. Much work remains for grassroots smartphone apps to be competitive with incumbent global platforms; but perhaps now this could be achieved with some AI-months rather than with incalculably-many person-years.

CCS Concepts

• **Computer systems organization** → **Peer-to-peer architectures**; • **Computing methodologies** → **Multi-agent systems**; • **Human-centered computing** → *Social networking sites*; • **Security and privacy** → *Digital signatures*; • **Theory of computation** → *Distributed computing models*; • **Applied computing** → *Law*; • **Social and professional topics** → *Computing / technology policy*.

Keywords

Grassroots Platforms, Grassroots Social Contracts, AI Risks, AI Agents, Accountability, Deepfakes, Digital Provenance, Digital Speech Acts, Genuine Digital Identity, Social Networks, Volition-Guarded Transactions, Abstraction Cascade, Smartphone Apps

TL;DR

Those who own and control the Web's social and economic platforms are now ceding their control to AI: of the algorithms, of who may join, and of what circulates. Hence, the majority of humanity now faces AI control of global platforms, unaccountable AI agents posing as people, and unattributed deepfakes. We propose a Web owned, operated and governed by the people who use it, with no "others" who control it and could cede their control to AI; AI agents join only under operators accountable for their operation and developers accountable for their defects; and every circulating item carries digital provenance, identifying its author and those who forwarded it, so a deepfake loses the protection of anonymity by having someone accountable for it.

1 Introduction

AI's inflection point is a fountain of alarming risks [2, 3, 20, 52, 56]. Digital platforms in particular are vulnerable to three major AI risks. Those in control of global platforms are ceding control to AI to maximise the value extracted from their users: of the algorithms, amplifying commercial surveillance [59]; of membership, by letting AI agents masquerade as people [15, 31]; and of content, by letting AI generate deepfakes with no one held accountable [28].

(1) **AI control of global platforms**—social and economic—used by the majority of humanity, resulting in the majority of humanity becoming vulnerable to AI control. Platform owners may delegate the operation of their platforms to AI, giving AI systems power over the social and economic lives of their participants and compounding existing concentrations of power, surveillance and manipulation [3, 10, 51, 58]; the participants are then subject to AI control even when the AI faithfully pursues its owners' objectives, and the owners may later be unable

to reclaim the control they delegated [2]. Kulveit et al. [24] analyse the broader possibility that successive decisions to adopt AI gradually erode human influence over societal systems.

(2) **Unaccountable AI agents** operating anonymously on social and economic digital platforms, side-by-side with humans, possibly masquerading as humans, serving unknown masters with unknown goals, operating under unknown restraints without accountability or culpability. Social bots already take part in social networks and impersonate people [15, 31]. Proposals to establish, privately, that a participant is a person [1] and to make an AI agent's activity attributable through identifiers, monitoring and logs [7, 51] address two different questions: whether a person is participating, and who authorised an agent's acts and is answerable for them. Attribution must also be related to actual authority and control [34]: an identifiable person may otherwise be made to bear responsibility for a system over which they had little control, Elish's moral crumple zone [13].

(3) **Unattributed deepfakes** generated by anonymous AI, with people unable to distinguish them from genuine content and with their producers evading accountability and culpability. Untrained observers distinguish synthetic faces from real ones at about chance level [30], and a deceptive political deepfake raises uncertainty and lowers trust in news on social media [55]; Mirsky and Lee survey the creation and detection of deepfakes [28], and Chesney and Citron their consequences for privacy, democracy and national security [9]. Detection estimates whether content is synthetic; provenance records who produced, endorsed or forwarded it [8], and supports accountability without establishing that the content is true [8, 18].

Two further AI risks are not addressed here: harms to people using AI, through erroneous advice, manipulation, overreliance or emotional dependence [2, 14, 16, 56], and harms people cause with abilities amplified by AI, such as fraud, harassment and cyberattacks [2, 5, 27].

Here, we propose to corral these AI risks with grassroots social contracts [6, 17], a digital incarnation of a two-centuries old notion of Pierre-Joseph Proudhon [33, Sixth Study]. Proudhon envisioned a social contract that can be freely adopted by any group of people—one could "make a contract with all, as … with some"—where groups bound by the same social contract can union by mutual consent, possibly into a single global union—"each group of citizens … formed by a like contract … could thereafter, and always by a similar contract, agree with every and all other groups". Grassroots social contracts are realised by grassroots platforms [26, 38, 40, 45]: smartphone-based person-to-person digital platforms owned, operated and governed by their participants. A grassroots platform is constituted from its participants' smartphones and the grassroots app each participant runs, and requires no global resource but the network: no hardwired servers, no global authority, and nobody's permission to start. Global platforms, centralised like Facebook or decentralised like Bitcoin and Ethereum [29, 57], can have only a single instance, as two instances would clash over the global resources—domain name, port number, or boot nodes—hardwired into their code, and from the perspective of ordinary people all of them are monolithic entities owned, operated and governed by others [41]. Fundamental grassroots platforms include the grassroots social graph [12, 17, 39], grassroots social networks [39], child-safe social networks [43], and grassroots currencies [25, 37, 44, 48], with higher-level platforms providing for democratic digital communities [19, 21, 50, 53] and "sharing-economy" cooperatives (in preparation).

We developed, with AI, a methodology to compile grassroots social contracts to working smartphone apps: an abstraction cascade from the contract to the app (Section 2), its correctness proved from the transactions down to GLP and the app built and tested, each layer coded by AI from the paper that describes it [11, 17, 22, 26, 42, 46, 49].

In this paper, we present grassroots social contracts for a social graph with accountable AI agents (Section 3) and for a social network with digital provenance (Section 4), and working smartphone apps that implement the two (Figures 4 and 5), and show how they corral these major AI risks:

(1) **AI control of global platforms** is mitigated by grassroots platforms by design, as, unlike global platforms, they are not controlled by a third party who may then cede control to AI. A grassroots platform has no servers, no operators and no global resources, so there is no part of it whose capture is control of the platform, and to control it an AI must control the smartphones of its participants, one by one [38, 40, 41].

(2) **Unaccountable AI agents** are excluded, as AI agents are introduced to the grassroots social graph with accountability: similarly to cars, their manufacturers are known and are accountable for manufacturing defects, and their operators are known and are accountable for reckless operation. A person takes part in a grassroots platform only with agents they met or were introduced to by an agent who knows them [17, 32, 35, 39], so an AI agent joins only under a person, its principal, who attests that it runs code signed by its developer and is accountable for operating it recklessly, the developer being accountable for its defects; every act of the AI agent names the two and is signed and non-repudiable [6, 12, 18]. The contract shares its core with the child-safe social network [43], whose wards are children and whose guardians are their parents.

(3) **Unattributed deepfakes** cannot circulate in grassroots social networks: every item is a digital speech act [18], signed by its author and carried along the edges of the social graph [17, 39], and signed over by each agent that forwards it, who warrants that they know personally the agent they received it from; so creators and forwarders of harmful content are accountable and culpable, and people can tell content with provenance from content without it. Some clauses of a grassroots social contract are enforced by a correct protocol, and others leave non-repudiable evidence for recourse when breached [17].

Grassroots platforms are presently a concept with mathematical and computational theory [17, 26, 38, 41, 45], working prototypes [11, 22, 46], and tools for their large-scale simulation and testing [47]. We have developed grassroots theory for a decade but could not muster the resources needed to turn it into competitive software, given the not-for-profit, common-good nature of our project. Now, AI is piercing through this conundrum: our scientific papers serve as the evolving interface between us humans and AI, which codes from them [46, 49]. Much work remains for grassroots smartphone apps to be competitive with incumbent

global platforms; but perhaps now this could be achieved with some AI-months rather than with incalculably-many person-years.

Section 2 explains how a grassroots social contract becomes an app, and presents the grassroots social graph and the grassroots social network with their apps. Section 3 presents the contract with accountable AI agents and Section 4 the contract with digital provenance, each with what it excludes and what it makes answerable. Section 5 reviews related work and Section 6 concludes. Appendix A recalls the framework, Appendix B lists the transactions, and Appendix C defines contracts with wards, an extension of the speech-act schema language of [17], and proves that the protocol realising one is volitionally grassroots, this contract's among them.

2 Grassroots Social Contracts

A grassroots social contract [6, 17] is a few clauses in plain language, each describing an act—who takes it, who must agree to it, and what it changes—or binding the agents without describing one. Befriending is an act of two persons, both of whom must agree; ending a friendship is an act of one. The contract is the whole of the platform's design: there is no server, no operator and no account, only the persons who adopted the contract and their smartphones [38, 40]. Each clause opens with its legal modality [17]: ■ *enforced*: no act of agents can breach it; ◆ *attested* by the signatures of the actors: an agent may breach it, and the contract records the breach; ○ *undertaken* by the actors: its performance lies outside the contract, which records only the undertaking. A contract ends with its *pragmas*, which the app built from it follows: the first line names a tab of the app and what it lists, a row showing the atoms of the state naming its item, and each other line names a clause by its title, then the agent proposing, a button for each act they propose, bearing its act's name, and any field they fill, and a line for each other agent asked, their name and the question put to them [17, 22]. We show the pragmas of GAI (Section 3) and GSNP (Section 4); the pragmas of every contract are in the repository https://github.com/EShapiro2/GLP, and GKK's, GSG's and GSN's in [17].

From the contract to the app. The clauses are written as speech-act schemas [17], which name the roles of an act, which of them must will it, and what each role's state must hold before and comes to hold after; the schemas compile into volition-guarded multiagent atomic transactions [26], each a change to the states of its participants that occurs only if the persons guarding it are willing; the transactions are realised by communicating volitional agents, whose only acts between two agents are discovery and message delivery, and then by a volition-guarded Grassroots Logic Program (vGLP) [22], which is transformed into the platform's user interface and compiled to Grassroots Logic Programs (GLP) [42, 49], implemented in Dart and Flutter [46] and run as a mini-app of the Grassroots Super-App on a person's smartphone [11] (Figure 1). Correctness is proved from the transactions down to GLP [22, 26]; the app is built and tested. Each layer is coded by AI from the paper that describes it; a gap found in coding is repaired in the paper first, and the code follows [46].

What a person sees. Volition elicitation is the rationale for the user interface design [22]. An act a person must will appears on their phone as a question, with the answers the contract allows [22, 26]: Bob is asked whether to be Alice's friend, and answers Accept or Decline. A person's phone holds their own state and nothing else [38–40], their friends, their groups, the messages they sent and received, and draws what it shows from that state, by widgets the grassroots app declares for the platform's questions and for the person's state [11, 22]. A person's phone talks to the phones of the agents they know or come to know, and to nobody else [11, 12, 39]. The screens of Figures 2 and 3 are screenshots of the GSG and GSN mini-apps, and those of Figures 4 and 5 of the GSNP mini-app, each compiled from its social contract down the abstraction cascade [17, 22, 26, 42, 46, 49]. One instance of the grassroots app [11] on an iPhone simulator runs the mini-app of every agent [11, 46]; the agents communicate over channels within it [42], and a screenshot shows one person's screen. All source code and all artefacts, including the social contracts and the artefacts the abstraction cascade produces from them, are in the public repository https://github.com/EShapiro2/GLP.

2.1 Grassroots social contracts, platforms and apps

Almost two centuries ago Pierre-Joseph Proudhon envisioned a social contract [33, Fourth Study] that is (1) an agreement of man with man; (2) reciprocal; (3) imposing no obligation upon the parties except that which results from their personal promise; (4) subject to no external authority; (5) freely accepted and signed by all the participants; and (6) of the nature of a contract of exchange. *Grassroots social contracts* are social contracts meeting these six conditions and two more: (7) people are free to deal with each other; and (8) there is no external register of people, so people can come to know each other either directly or through people they already know [17].

A *digital social contract* is one realised by software the parties run [6]. We call the mathematical object a *protocol* and the software implementing it a *platform*. A protocol is *grassroots* when its instances are independent of one another and of every global resource but the net, and can always coalesce, possibly into a single global instance [38, 40, 41], and a *grassroots platform* is the software implementing a grassroots protocol [17].

A grassroots platform is constituted from its participants' smartphones and the grassroots app each participant runs [38, 40]; a person runs one app, and each platform is a mini-app of it [22]. Participating in a grassroots platform requires adopting the grassroots social contract that defines it, as participating in a global platform requires adopting an End-User Licence Agreement (EULA). A grassroots social contract is mutual and symmetric among all participants; an EULA is between each person and the platform operator. To join a grassroots platform, a person signs a one-sided undertaking to keep its grassroots social contract towards all its other participants, and presents it to every participant they interact with, so every participant has signed it [17].

2.2 The grassroots social graph

A *person* is a legally culpable entity: a natural person of age, also referred to as an *adult*, or a legal person of good standing, such as a business. A person takes the pronoun 'they'. An *agent* is a person, a *child*, a natural person not of age, or an *AI agent*, a program that holds a key pair of its own and takes acts of the contract with it,

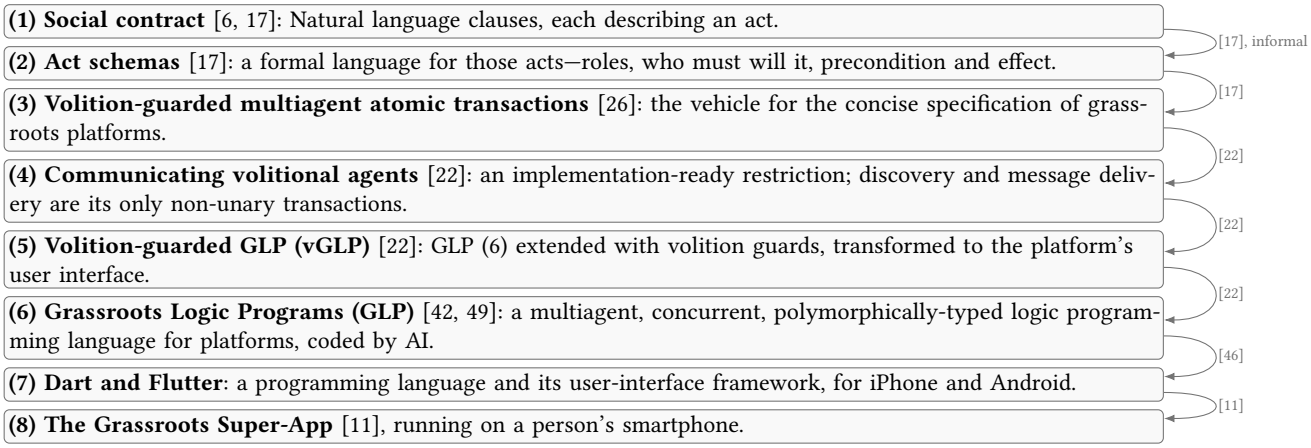


Figure 1: The abstraction cascade, from a grassroots social contract to a running app. Each arrow but the first, an informal translation, represents an implementation, with the reference defining it.

and takes the pronoun 'it'; this paper treats AI agents, and the child-safe social network [43] children. Every contract extends GKK, the base contract: a person establishes their identity by choosing a key pair, and *knows* another agent when they hold that agent's public key [17].

GKK: Grassroots Keys and Knowledge [17]

Alice, Bob and Carol are persons.

1. ○ *Identity.* Alice may act once to establish their identity by choosing their unique key pair. They make themselves known by the public key, sign their acts with the private key, and are accountable for every act signed with it.
2. ■ *Acquaintance.* Alice and Bob may come to know each other by an act of both.
3. ■ *Introduction.* If Alice and Bob know each other and Alice knows Carol, then an act of Alice and Bob can introduce Carol to Bob, following which Bob knows Carol.

A contract that extends another adds clauses to it and cites its clauses whole; GSG extends GKK.

GSG: Grassroots Social Graph [17]

Alice and Bob are persons.

1. ◆ *Friendship.* Alice and Bob, who know each other personally and are not friends, may jointly decide to become friends by an act of both.
2. ■ *Ending a friendship.* Alice may end their friendship with Bob by their own act.

GSG is the grassroots social graph [17, 39]. The GSG app is a mini-app of the grassroots app, compiled down the cascade from GSG; Figure 2 shows it realising GSG:1.

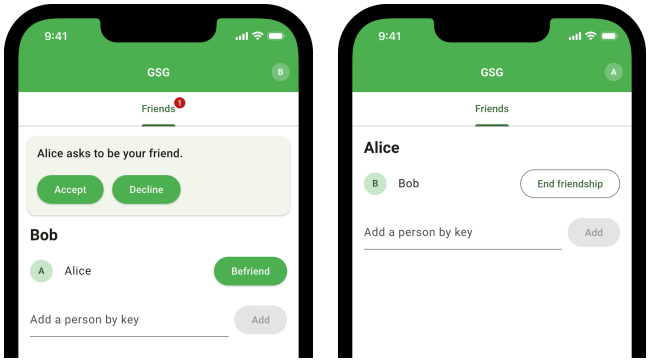


Figure 2: The GSG app: Alice asks to be Bob's friend, and Bob answers Accept or Decline (GSG:1); Alice's friends after Bob accepts.

2.3 The grassroots social network

A *relation* is a friendship or the membership of a group.

GSN: Grassroots Social Network [17]

Alice and Bob are persons.

1. ■ *Message.* Alice may send a message to their friend Bob.
2. ■ *Group.* Alice may create a group, and is then its leader and first member.
3. ■ *Joining.* Alice, the leader of a group, may add their friend Bob to it, by an act of both.
4. ■ *Leaving.* Bob, a member of a group, may leave it by their own act; when the leader leaves, the group ends. A member of a group who is not a friend of its leader is removed from it.
5. ■ *Group message.* Bob, a member of a group, may send a message to the group, and the leader delivers it to every current member.

The GSN app is a mini-app of the grassroots app, compiled down the cascade from GSN; Figure 3 shows it realising GSN:2, GSN:3 and GSN:5.

3 A Social Graph with Accountable AI Agents

This section presents a grassroots social contract [6, 17] for a social graph in which AI agents take part, each with a principal and a developer accountable for it, and shows which parts of the second risk of Section 1 the contract excludes and which it makes answerable. The contract has three parts: the grassroots social graph with GKK (Section 2.2) and GnW, guardians and wards, both shared with the child-safe social network [43], whose third part is for parents and

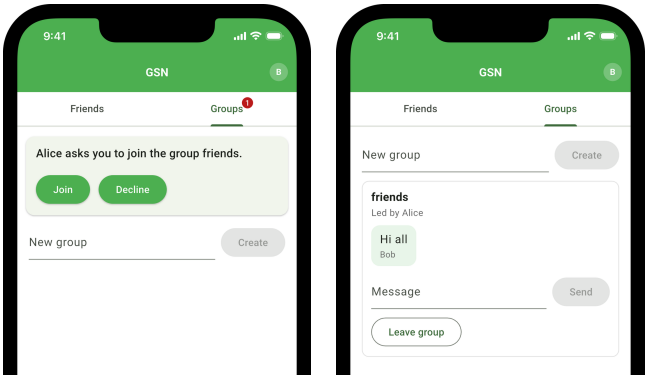


Figure 3: The GSN app, on Bob's phone: Alice asks Bob to join her group *friends*, and Bob answers Join or Decline (GSN:3); the group *friends*, led by Alice (GSN:2), with Bob's message to the group as Alice delivered it to him (GSN:5).

children; and the AI-agents extension, the third part here; Section 4 adds the social network of Section 2.3 and its provenance.

The running example is Alice's AI agent Alf befriending Bob, Acme Ltd being Alf's developer. It rests on one clause of each modality: ■ Alf may befriend Bob by an act of Alf, Bob and Alice (GnW:2); ◆ Alice, when creating Alf, attests that it runs code signed by its developer, Acme Ltd (GAI:1); ○ Acme Ltd is accountable for Alf's defects, and Alice for operating it recklessly (GAI:3).

3.1 A Social Contract for Guardians and Wards

GnW covers both kinds of ward, AI agents and children.

GnW: Guardians 'n' Wards

Alice and Dave are persons, Bob an agent, and Carol a ward.

1. ■ *Guardianship.* Alice may act to create a *ward*, Carol, to be operated by a non-person agent, and establish Carol's identity by choosing Carol's key pair, and Alice is then the *creator* and a *guardian* of Carol, as well as their friend.
2. ■ *Knowing and relations.* Carol, a ward of Alice, comes to know Bob, and enters a relation with Bob, by an act of both, of Alice, and of Dave, a guardian of Bob, if Bob is a ward, Alice and Dave possibly one person.
3. ■ *Ending a relation.* Bob, who is in a relation with a ward, Carol, may act to end it, and so may Alice, a guardian of Carol.
4. ■ *Retiring.* The creator of Carol may retire Carol, which ends every guardianship of Carol, after which Carol takes no act.

An AI agent is made by a person, its *developer*, and is operated by a person, its *principal*. Accountability for an AI agent is divided between the two as it is for a car between its manufacturer and its driver: the manufacturer is accountable for manufacturing defects, and the driver for reckless driving. So the developer signs the code of the AI agent, its principal attests on creating it that it runs that code by that developer, so that every agent the AI agent knows holds the attestation and learns who is accountable for it.

GAI: Accountable AI Agents

Alice is a person and Alf an AI agent.

1. ◆ *Developer.* The *developer* of an AI agent, Alf, is the person who signed Alf's code. Alice, who creates Alf, attests that Alf runs code signed by its developer.
2. ■ *Principal.* Alice, the guardian of Alf, is Alf's *principal*. Every act of Alf names its principal and its developer and carries its principal's attestation.
3. ○ *Accountability.* The developer of Alf is accountable for Alf's defects, and Alf's principal for operating it recklessly.
4. ■ *Retiring.* Alice retires their AI agent Alf by ending their friendship with it.

Pragmas

AI agents: `those Alice is principal of.`
Guardianship: `Alice [Enter] <name>`
  Carol: `Alice asks to be your principal.`
Developer: `Alice [Sign] <text>`
Knowing and relations: `Alice [Agent acquaint] <person>`
  Carol: `Alice asks that you come to know Bob.`
  Bob: `Alice asks that you come to know their AI agent Carol.`
  `Alice [Agents acquaint] <person> <person>`
  Carol: `Alice asks that you come to know Bob.`
  Bob: `Alice asks that you come to know Carol.`
  Dave: `Alice asks that Carol and your AI agent Bob come to know each other.`
  `Alice [Agent befriend] <person>`
  Carol: `Alice asks that you and Bob befriend.`
  Bob: `Alice asks that you and their AI agent Carol befriend.`
  `Alice [Agents befriend] <person> <person>`
  Carol: `Alice asks that you and Bob befriend.`
  Bob: `Alice asks that you and Carol befriend.`
  Dave: `Alice asks that Carol and your AI agent Bob befriend.`
Ending a relation: `Alice [Agent unfriend] <person>`
Retiring: `Alice [Retire]`

In GAI the guardians are principals and the wards their AI agents. The principal's will is given once, when they create the AI agent; after that the AI agent acts on its own, and the principal takes part in each acquaintance of the AI agent and in each friendship it enters (GnW:2): an AI agent comes to know an agent, and befriends one, only with its principal. The principal signs the attestation once too, when they create the AI agent, and the AI agent presents it to each agent it comes to know, who records it (GAI:2). A principal may be a legal person, so a company's AI agents act under the company. A guardian signs the undertaking to keep the contract for their ward [17].

Figure 4 shows the contract on a person's phone, in the GSNP app of Section 4: a person creates an AI agent, attesting that it runs code signed by its developer, approves each acquaintance and each friendship of the AI agent, and may retire it; every agent an AI agent knows sees its principal and its developer.

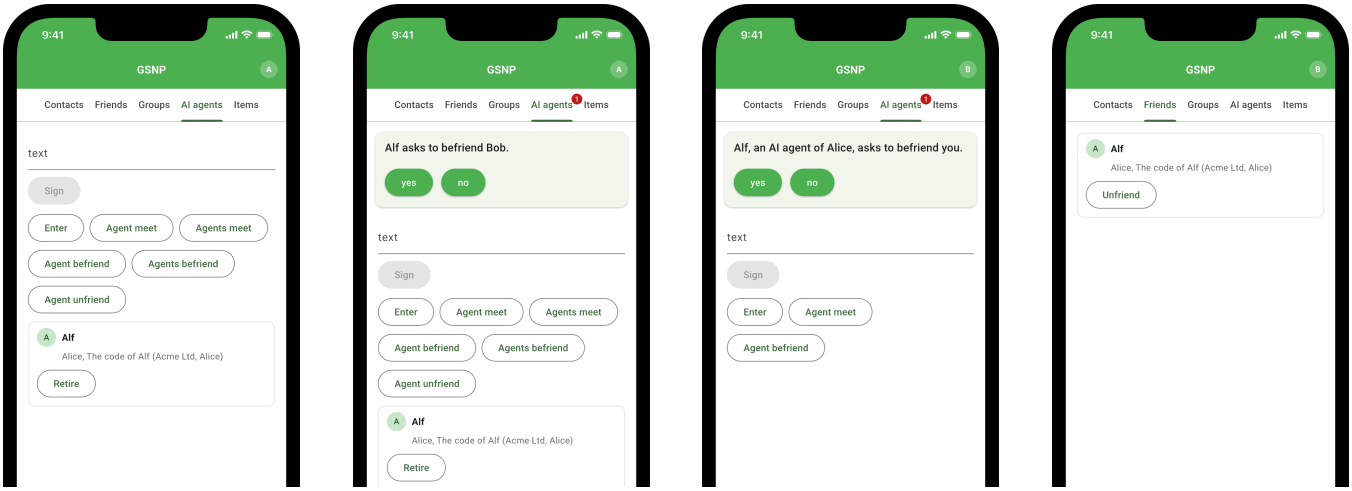


Figure 4: Accountable AI agents, in the GSNP app: Alice's AI agent Alf, its code signed by Acme Ltd and attested by her, and its Retire button (GnW:1, GAI:1 and GAI:4); Alice is asked to approve Alf befriending Bob (GnW:2); Bob is asked to befriend Alf, an AI agent of Alice (GAI:2); Bob's friends, Alf among them with its principal and its developer.

3.2 The transactions

The contract is realised by volition-guarded multiagent atomic transactions [26]; the speech-act schemas of the contract (Appendix B) determine them by the compilation of [17]. We present the transactions for knowing and friendship, of GKK, GSG, GnW and GAI, with the signing, messaging and forwarding of Section 4; a ward's membership of a group, the other relation of GnW:2, is realised as its friendship is, by an act of the leader, the ward and its guardian, and its transactions are left out. We assume a potentially infinite set of *agents* Π, but consider only finite subsets of it, so when referring to a particular set of agents $P \subset \Pi$ we assume P to be nonempty and finite; here an agent is a person or an AI agent, with its machine. We use $\subset$ for the strict subset relation and $\subseteq$ when equality is also possible, S^P for the set of all total functions from P to S, and c_p for the value of $c \in S^P$ at $p \in P$.

Definition 3.1 (Machine State, Configuration, Transaction, Volition–Guarded Multiagent Atomic Transaction [26]). Given an arbitrary set S of *machine states*, with a designated *initial state* $s_0 \in S$, and agents $Q \subset \Pi$, a *machine configuration* over Q is a member of S^Q, and a *machine transaction* over *participants* Q is a pair $c \to c' \in (S^Q)^2$ such that $c \neq c'$; it is *unary* if $|Q| = 1$. Given such a machine transaction t, a *volition-guarded multiagent atomic transaction* over t—henceforth, *volition-guarded transaction*—is a pair (t, Q') where $Q' \subseteq Q$ are its *guards*.

A volition-guarded transaction can be carried out only if its guards are willing; its participants include those whose state changes and those whose state is a precondition and does not change. The rest of the framework of [26]—agent states, runs, the protocol over a set of transactions, and what makes it oblivious, interactive, grassroots and volitionally grassroots—is recalled in Appendix A.

The local state. The speech-act schemas determine the local state of an agent p, each component standing for the atoms of a predicate, all initially empty or zero (Table 1, Appendix B). An agent is a *person* when $a_p = 0$ and an *AI agent* when $a_p = 1$, and the *principal* of an AI agent r is the p with $\rho_r = \{p\}$ and its *attestation* the y with $\gamma_r = \{y\}$. Every transaction requires $\iota = 1$ at each of its participants, except Establish, and Enter at the AI agent, which require $\iota = 0$ there, and $\theta = 0$ at each of its guards; the transactions below leave these requirements out, and all components not explicitly updated are preserved.

Example 3.2 (An AI agent coming to know a person, and befriending them). The two transactions below realise GnW:2 for an AI agent r with principal p and a person q: r and q come to know each other by an act of both and of p, q recording the attestation y of r with r and p (GAI:2), and then may become friends by an act of the three; p may end the friendship by Agent unfriend, and the two friends by Unfriend (Appendix B). The participants of each are r, q and p, and each is guarded by all three.

(3) Volition-Guarded Multiagent Atomic Transactions

(1) **Agent acquaint**$(r, q, p; y)$, where $a_r = 1$, $\rho_r = \{p\}$, $y \in \gamma_r$, $(r, p, y) \in \epsilon_p$ and $a_q = 0$:
$\nu'_r := \nu_r \uplus \{q\}$, $\nu'_q := \nu_q \uplus \{r\}$ and $\epsilon'_q := \epsilon_q \cup \{(r, p, y)\}$, guarded by $\{r, q, p\}$.

(2) **Agent befriend**(r, q, p), where $a_r = 1$, $\rho_r = \{p\}$, $q \in \nu_r$, $a_q = 0$, $r \in \nu_q$ and $q \notin c_r$:
$c'_r := c_r \cup \{q\}$ and $c'_q := c_q \cup \{r\}$, guarded by $\{r, q, p\}$.

For example, let Alice and Bob, $a, b \in \Pi$, be persons and Alf, $r \in \Pi$, Alice's AI agent. Alice and Bob establish their identities, come to know each other and befriend; Alice signs k, the code of Alf signed by its developer, and creates Alf with the attestation $y = sgn(k, a)$, k signed over by her; Alf comes to know Bob, with Alice willing, and Bob records that Alf is Alice's AI agent, with her attestation; Alf befriends Bob, with Alice willing; Alf signs an item x and sends it to Bob; Alice retires Alf; and Bob ends his friendship with Alf. The components that change are in Table 2 (Appendix B), Bob's record and the item remaining with him after the friendship ends.

3.3 Properties

This section states what the transactions enforce about an AI agent and its principal, with the argument for it, and that the protocol is volitionally grassroots, proved in Appendix C.

The standing of a principal. Consider any configuration reached by a run of the protocol (Appendix B) and an AI agent r in it, $a_r = 1$, with $\rho_r = \{p\}$.

(1) *Guardianship.* The identity of r was established by Enter$(p, r; k)$, guarded by p, for an item k of p, and $\rho_r = \{p\}$ and $\gamma_r = \{sgn(k, p)\}$ in every configuration since; and $p \in c_r$ if and only if $\theta_r = 0$.
(2) *Whom r knows and befriends.* Every $s \in v_r$ other than p came to know r by an Agent acquaint or an Agents acquaint guarded by p, and $(r, p, y) \in \epsilon_s$ for the attestation y of r; every $s \in c_r$ other than p is in v_r and became a friend of r by an Agent befriend or an Agents befriend guarded by p.
(3) *The power to end each.* For every $s \in c_r$ other than p, Agent unfriend(r, s, p) is machine-enabled and guarded by p; while $\theta_r = 0$, Retire(p, r), which ends the friendship of r and p, is machine-enabled and guarded by p.
(4) *Retirement.* If $\theta_r = 1$ then r has left, and $\rho_r = \{p\}$.

Each holds by inspection of the transactions. (1) Only Enter sets $a := 1$, ρ and γ, and no transaction changes them, so $a_r = 1$, $\rho_r = \{p\}$ and $\gamma_r = \{sgn(k, p)\}$ were set by one Enter$(p, r; k)$, which requires $k \in h_p$ and $\iota_r = 0$ and is guarded by p; before it r had the initial state, as every other transaction requires $\iota = 1$ at its participants, and Enter sets $c'_r := \{p\}$. While $\theta_r = 0$, p stays in c_r: Unfriend excludes a principal and an AI agent of the agent ending the friendship, Agent unfriend excludes the principal, and Retire, the one other transaction removing from c_r, sets $\theta_r := 1$; no transaction sets $\theta := 0$, and every transaction adding to c_r is guarded by r and so requires $\theta_r = 0$. (2) The transactions adding to v_r are Acquaint and Introduce, which require $a = 0$ at the agent coming to know; Enter, which adds p; and Agent acquaint and Agents acquaint, which require $\rho_r = \{p\}$, $y \in \gamma_r$ and $(r, p, y) \in \epsilon_p$, which Enter adds, are guarded by p, and add (r, p, y) at the agent coming to know r; no transaction removes a triple from ϵ. The transactions adding to c_r are Befriend, which requires $a_r = 0$; Enter, which adds p; and Agent befriend and Agents befriend, which require $\rho_r = \{p\}$ and the new friend in v_r, and are guarded by p. (3) The requirements of Agent unfriend(r, s, p) and of Retire(p, r) hold—$p \in c_r$ by (1), and $\theta_p = 0$, since Retire, the one transaction setting $\theta := 1$, requires $a = 1$ there and $a_p = 0$—and each is guarded by p. (4) Every transaction guarded by r requires $\theta_r = 0$, which no transaction unsets, so once $\theta_r = 1$ no transaction r guards is machine-enabled: r has left, and $\rho_r = \{p\}$ by (1).

Part (1) is why what the contract says of an AI agent is enforced rather than attested: an agent the contract records as an AI agent of p cannot cease to be one by any act, its own or another's, and part (4) is the one way it leaves. Retirement ends the guardianship (GnW:4): the retired AI agent takes no act; the record of its principal stays, for its past acts.

Grassroots. The protocol realising the contract is volitionally grassroots (Proposition C.22, Appendix C, by the theorem for contracts with wards proved there): two persons can always befriend, an AI agent that is not retired can always befriend a person or another AI agent with its principal willing, and through befriending any agent can come to interact with any other; two groups that have formed apart, each with an agent that has not left, can always coalesce, and only by an act an agent of each wills. A retired AI agent has left.

3.4 What the contract excludes and what it makes answerable

A clause is enforced, attested or undertaken (Section 2) [17]: no act of agents can breach an enforced clause; a breach of an attested clause leaves non-repudiable evidence against the agent in breach in another agent's hands; and an undertaken clause is kept outside the contract, the agent's signed undertaking to keep the contract, held by every agent they interact with, being its record, so a breach of an attested or an undertaken clause can be taken to court. The second risk of Section 1 has five parts, and we take them in turn.

Anonymity. A person signs their acts with the private key of their genuine digital identity (GKK:1); an AI agent's key pair is chosen by its principal (GnW:1), and every act of an AI agent is signed with its private key and names its principal (GAI:2). Both are enforced: an agent without an identity has no act of the contract available to it, and an act of the contract is a digital speech act, an utterance signed with the key of the agent taking it [17, 18].

Masquerading as a person. An AI agent is recorded as one on its creation (GnW:1 and GAI:2), is a ward from its first act and remains one (Section 3.3(1)), and every acquaintance and every friendship of it is an act of its principal (GnW:2), so every agent an AI agent knows, and so every agent it interacts with, holds a record naming it as an AI agent and naming its principal, with its principal's attestation (Section 3.3(2)): enforced. What the contract cannot determine is whether an agent that chose its own key pair is a person: no act of the contract can tell a person's act from a program's taken under that person's identity. Such a program is not an AI agent of the contract; its acts are signed with the person's private key and are that person's acts, for which the person is accountable (GKK:1). A person's accountability for the acts signed with their private key is undertaken: the person's signed undertaking, held by every agent they interact with, is its record.

Unknown masters. The principal of an AI agent is a person with a genuine digital identity, recorded in the AI agent, named in each of its acts and taking part in each of its acquaintances and friendships (GnW:2 and GAI:2): enforced.

Unknown goals and restraints. The contract does not determine what an AI agent is for or what its principal lets it do. It determines who is accountable for it: its developer for its defects, and its principal for operating it recklessly (GAI:3). The clause is undertaken, the principal's signed undertaking being held by every agent their AI agent comes to know. GAI:1 is attested: the principal signs the attestation when they create the AI agent, every agent the AI agent knows holds it (Section 3.3(1) and (2)), and an attestation of code the AI agent does not run is evidence against the principal that they cannot repudiate.

Accountability and culpability. Every act of an AI agent is signed with its key, names its principal and carries its principal's attestation (GAI:2), and the agent it is addressed to holds it: non-repudiable evidence attributing the act to the AI agent and the AI agent to its

principal [18]. Harm done by an AI agent under this contract is harm done by an identified agent, whose principal is accountable for operating it recklessly and whose developer is accountable for its defects (GAI:3).

Hence on a platform realising this contract there are no anonymous agents, natural or computational, and so there are no anonymous AI agents: an AI agent acts under an identity of its own, recorded as an AI agent and naming its principal, or a program acts under a person's identity and that person is accountable for its acts.

4 A Social Network with Digital Provenance

This section extends the contract of Section 3 by GSNP, the grassroots social network with provenance [39], which extends GSN (Section 2.3) by three clauses: an agent signs the content they create, forwards content they received with its signatures and their own, warranting that they know personally the agent they received it from, and discloses content they received only by forwarding it. It then shows which parts of the third risk of Section 1 the contract excludes and which it makes answerable. An *item* is a speech act: content signed by its author [18], and, when forwarded, signed over by each agent that forwarded it, in order. GSNP's speech-act schemas are in the repository, and the protocol realising it is volitionally grassroots (Proposition C.21, Appendix C).

4.1 Extending the contract with provenance

GSNP: Grassroots Social Network with Provenance

Alice and Bob are agents.

1. ■ *Authorship.* Alice signs the content they create.
2. ◆ *Forwarding.* Alice forwards content they received by sending it to Bob with the signatures it has and adding their own, and warrants that they know personally the agent they received it from.
3. ○ *Confidence.* Alice shall disclose content they received only by forwarding it.

Pragmas

Items: `the items Alice holds.`
Message: `Alice [Message] <item>`
Group message: `Bob [Post] <item>`
`Alice [Post own] <item>`
Forwarding: `Alice [Forward]`
Alice: `You warrant that you know personally the agent you received this from.`
`Bob [Post forward]`
Bob: `You warrant that you know personally the agent you received this from.`

GSNP adds provenance to the messages of GSN: a message to a friend (GSN:1) or to a group (GSN:5) carries an item. GSNP:1 and GSNP:2 describe acts: signing adds the item to its author's holdings; a message carries an item its sender authored, as they signed it; forwarding adds to the friend's holdings an item the forwarder received, signed over by the forwarder; and a leader delivers a member's item to the group by forwarding it, signed over by the leader; so an item in an agent's hands is signed by its author and by every agent that forwarded it, in order, and carries its provenance inside it. The warranty in GSNP:2 describes no act of its own, and neither does GSNP:3; both bind the agents. An agent holds an item by signing it or by receiving it from a friend, the leader of a group being a friend of each of its members (GSN:3 and GSN:4), and in no other way: content that reaches a person otherwise is not an item of the platform. Accountability for the items of an AI agent is as in GAI:3.

The GSNP app is the GSN app with provenance and with the AI agents of Section 3 (Figure 5): every item shows its author and every agent that forwarded it, and forwarding asks for the warranty of GSNP:2.

4.2 What the contract excludes and what it makes answerable

The third risk of Section 1 has three parts, and we take them in turn, with the three modalities of Section 3.4.

Anonymous content. Every item is signed by its author (GSNP:1), an agent with a genuine digital identity (GKK:1), and an agent holds an item only by signing it or by receiving it from a friend, who authored it or signed it over (GSNP:2): enforced. Every item an agent holds carries the signatures of its author and of each agent it passed through, and content signed by nobody is not an item of the platform. Whether the content was produced by a person or by an AI is not what the signatures say; who is accountable for it is.

Content people cannot tell from genuine content. The platform does not decide whether content is synthetic; it decides who signed it. An item authored by an AI agent is signed by the AI agent, which every agent it knows holds recorded as an AI agent of its principal (GnW:2 and GAI:2): a reader who knows the AI agent sees that the item is an AI agent's and whose; enforced. A reader who does not know the AI agent sees its signature and the forwarders', each of whom knows personally the agent they received it from (GSNP:2). An item a person signs is that person's, whatever produced it, and the person is accountable for it (GKK:1): a person who signs AI-generated content as their own is its author. An item that purports to be a person's and is not carries that person's signature or it does not, and a signature cannot be forged [18]; an item that depicts or quotes a person is its author's statement about them, and the author is accountable for it. So a reader of a grassroots social network tells content with provenance from the rest by its signatures, which every item carries, and not by its look.

Producers evading accountability. The signature of an item is non-repudiable evidence that its author produced it, held by every agent that holds the item, and each forwarder's signature over it is evidence that they passed it on: attested. The warranty of GSNP:2 is attested too: a forward by an agent who does not know personally the agent they received the item from makes it false, and the forward leaves the item, signed by the forwarder, in the friend's hands [17]. GSNP:3 is undertaken: no act of the contract breaches it, and the agent's signed undertaking is its record. The author of an item, a person or an AI agent with an accountable developer and principal (GAI:3), and each agent that forwarded it, are identified agents, and harm done by an item can be taken to court against them [17, 18].

Hence on a platform realising this contract there is no anonymous content: every item carries the signatures of its author, a

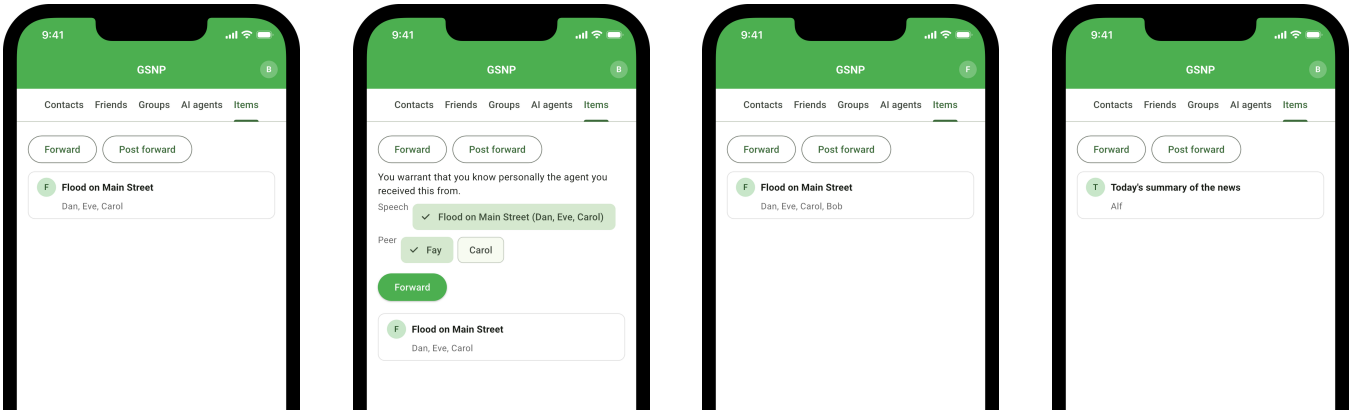


Figure 5: The GSNP app: on Bob's phone, an item with its author and the agents that forwarded it (GSNP:1 and GSNP:2); Bob forwarding it to Fay, with the warranty (GSNP:2); on Fay's phone, the item with Bob among its forwarders; on Bob's phone, an item authored by the AI agent Alf (GSNP:1).

person accountable for it or an AI agent whose principal and developer are accountable for it, and of every agent that forwarded it. Deepfake content may be produced outside the platform and signed into it, and it then has an author who is accountable for it; what the platform does not carry is content with no one behind it.

5 Related Work

AI risks. Taxonomies and surveys of AI risks include those of Weidinger et al. [56], Bommasani et al. [3] and Hendrycks et al. [20], the AI Risk Repository [52] and the International AI Safety Report [2]. Algorithmic governance and regulation [10, 58] and gradual disempowerment [24] bear on the first risk; personhood credentials [1], social bots [15], visibility into AI agents [7], the governance of agentic systems [51], moral crumple zones [13] and meaningful human control [34] on the second; deepfakes [9, 28, 30, 55] and reducing the risks of synthetic content [8] on the third; and AI assistants [16], chatbot use [14] and malicious use [5, 27] on the two risks not addressed here. We address the three through the organisation of the platform.

Grassroots platforms. Grassroots platforms [38, 40] are characterised against global platforms, centralised, decentralised and federated [41], the decentralised ones exemplified by Bitcoin [29] and Ethereum [57]. They are specified by atomic transactions [45] and volition-guarded multiagent atomic transactions [26], and defined by grassroots social contracts [17], digital social contracts [6] meeting Proudhon's conditions [33]. Their instances include grassroots social networks [39], currencies [37] and bonds [44], and a child-safe social network [43]. Their identities are genuine personal identifiers [35], their communities grow sybil-resiliently [32], their content is digital speech acts [18], and their participants recover from the loss or compromise of a key or a device with the help of their friends [12].

The cascade. A contract is compiled down to a phone through volition elicitation and vGLP [22], Grassroots Logic Programs [42], their types [49] and their implementation [46], and the Grassroots Super-App [11], in which each platform is a mini-app.

Person-to-person systems. Two deployed systems share the person-to-person architecture of grassroots platforms. In Secure Scuttlebutt [54] each participant publishes to a single-writer, append-only log and replicates the logs of the identities they choose, so data flows along the edges of a social graph, as in grassroots social networks [39]. Briar [4] synchronises messages directly between its users' devices, with no central server, and over Bluetooth or Wi-Fi when the Internet is down. Both are classified as grassroots by their essential agents, informally [41]. Local-first software [23] keeps the primary copy of a person's data on their own devices, with servers holding secondary copies, and merges concurrent changes by conflict-free replicated data types that sync over any channel, a peer-to-peer connection included; in a grassroots platform, the copies of a person's state are held by the state custodians they designate among their friends [12].

6 Conclusion

Grassroots platforms corral three AI risks of digital platforms. A grassroots platform has no part whose capture is control of it, so to control it an AI must control the smartphones of its participants, one by one. Under the contract of Section 3, every AI agent has a principal and a developer, every act of an AI agent names its principal and carries its principal's attestation that it runs code signed by its developer, and its principal and its developer are accountable for it. Under the contract of Section 4, every item carries the signatures of its author and of every agent that forwarded it.

References

[1] Steven Adler et al. 2024. Personhood credentials: Artificial intelligence and the value of privacy-preserving tools to distinguish who is real online. https://doi.org/10.48550/arXiv.2408.07892 arXiv:2408.07892
[2] Y. Bengio et al. 2026. *International AI Safety Report 2026.* DSIT 2026/001. Department for Science, Innovation and Technology. https://internationalaisafetyreport.org/publication/international-ai-safety-report-2026
[3] Rishi Bommasani et al. 2021. On the Opportunities and Risks of Foundation Models. arXiv:2108.07258. https://doi.org/10.48550/arXiv.2108.07258 arXiv:2108.07258
[4] Briar Project. 2018. Briar: Secure messaging, anywhere. https://briarproject.org.
[5] Miles Brundage et al. 2018. The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation. https://doi.org/10.48550/arXiv.1802.07228 arXiv:1802.07228
[6] Luca Cardelli et al. 2020. Digital social contracts: A foundation for an egalitarian and just digital society. In *CEUR Proceedings of the First International Forum on Digital and Democracy*, Vol. 2781. CEUR-WS, 51–60. https://ceur-ws.org/Vol-2781/paper5.pdf

[7] Alan Chan et al. 2024. Visibility into AI Agents. In *The 2024 ACM Conference on Fairness, Accountability, and Transparency*. Association for Computing Machinery, 958–973. https://doi.org/10.1145/3630106.3658948

[8] Bilva Chandra et al. 2024. *Reducing Risks Posed by Synthetic Content: An Overview of Technical Approaches to Digital Content Transparency*. NIST AI 100-4. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.100-4

[9] Robert Chesney and Danielle Keats Citron. 2019. Deep fakes: a looming challenge for privacy, democracy, and national security. *California Law Review* 107 (2019), 1753.

[10] John Danaher et al. 2017. Algorithmic governance: Developing a research agenda through the power of collective intelligence. *Big Data & Society* 4, 2 (2017). https://doi.org/10.1177/2053951717726554

[11] Ohad Eitan, Idit Keidar, and Ehud Shapiro. 2026. The Grassroots Super-App: A Secure Operating System for Grassroots Mini-Apps. arXiv:XXX:XXXXX. arXiv:XXX:XXXXX [cs.DC]

[12] Ohad Eitan, Idit Keidar, and Ehud Shapiro. 2026. Securing People and their Machines Against Major Faults (Full Version). *Proceedings of The 28th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS 2026), to appear* (2026). https://arxiv.org/abs/2607.02304 arXiv:2607.02304.

[13] Madeleine Clare Elish. 2019. Moral Crumple Zones: Cautionary Tales in Human-Robot Interaction. *Engaging Science, Technology, and Society* 5 (2019), 40–60. https://doi.org/10.17351/ests2019.260

[14] Cathy Mengying Fang et al. 2025. How AI and Human Behaviors Shape Psychosocial Effects of Extended Chatbot Use: A Longitudinal Randomized Controlled Study. https://doi.org/10.48550/arXiv.2503.17473 arXiv:2503.17473

[15] Emilio Ferrara et al. 2016. The Rise of Social Bots. *Commun. ACM* 59, 7 (2016), 96–104. https://doi.org/10.1145/2818717

[16] Iason Gabriel et al. 2024. The Ethics of Advanced AI Assistants. arXiv:2404.16244. https://doi.org/10.48550/arXiv.2404.16244 arXiv:2404.16244

[17] James Golike, Andy Lewis-Pye, and Ehud Shapiro. 2026. Grassroots Social Contracts: From Legal Text to an App (Full Version). *arXiv:2609.06231* (2026).

[18] James Golike and Ehud Shapiro. 2026. Digital Speech Acts Retain Control of Copyright with People, Not Platforms. *arXiv preprint arXiv:2606.19263* (2026). https://arxiv.org/abs/2606.19263

[19] Daniel Halpern, Ariel D Procaccia, Ehud Shapiro, and Nimrod Talmon. 2024. Federated Assemblies. *Proc AAAI 2025; arXiv preprint arXiv:2405.19129* (2024). https://arxiv.org/abs/2405.19129

[20] Dan Hendrycks, Mantas Mazeika, and Thomas Woodside. 2023. An Overview of Catastrophic AI Risks. arXiv:2306.12001. https://doi.org/10.48550/arXiv.2306.12001 arXiv:2306.12001

[21] Idit Keidar, Andrew Lewis-Pye, and Ehud Shapiro. 2025. Constitutional Consensus. *arXiv preprint arXiv:2505.19216* (2025). https://arxiv.org/abs/2505.19216

[22] Idit Keidar and Ehud Shapiro. 2026. Volition Elicitation: Operational Semantics for People and Their Machines. arXiv:2607.14138. arXiv:2607.14138 [cs.DC] https://arxiv.org/abs/2607.14138

[23] Martin Kleppmann, Adam Wiggins, Peter van Hardenberg, and Mark McGranaghan. 2019. Local-First Software: You Own Your Data, in spite of the Cloud. In *Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software (Onward! '19)*. 154–178. https://doi.org/10.1145/3359591.3359737

[24] Jan Kulveit et al. 2025. Gradual Disempowerment: Systemic Existential Risks from Incremental AI Development. arXiv:2501.16946. https://doi.org/10.48550/arXiv.2501.16946 arXiv:2501.16946

[25] Andrew Lewis-Pye, Oded Naor, and Ehud Shapiro. 2023. Grassroots Flash: A Payment System for Grassroots Cryptocurrencies. *arXiv preprint arXiv:2309.13191* (2023). https://arxiv.org/abs/2309.13191

[26] Andy Lewis-Pye and Ehud Shapiro. 2026. Volition-Guarded Multiagent Atomic Transactions: Describing People and their Machines. *arXiv preprint arXiv:2604.25596* (2026). https://arxiv.org/abs/2604.25596

[27] Nahema Marchal et al. 2024. Generative AI Misuse: A Taxonomy of Tactics and Insights from Real-World Data. arXiv:2406.13843. https://doi.org/10.48550/arXiv.2406.13843 arXiv:2406.13843

[28] Yisroel Mirsky and Wenke Lee. 2021. The Creation and Detection of Deepfakes: A Survey. *Comput. Surveys* 54, 1 (2021), 1–41. https://doi.org/10.1145/3425780

[29] Satoshi Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf

[30] Sophie J. Nightingale and Hany Farid. 2022. AI-synthesized faces are indistinguishable from real faces and more trustworthy. *Proceedings of the National Academy of Sciences* 119, 8 (2022), e2120481119. https://doi.org/10.1073/pnas.2120481119

[31] Joon Sung Park et al. 2023. Generative Agents: Interactive Simulacra of Human Behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST 2023)*. 1–22. https://doi.org/10.1145/3586183.3606763

[32] Ouri Poupko, Gal Shahaf, Ehud Shapiro, and Nimrod Talmon. 2021. Building a sybil-resilient digital community utilizing trust-graph connectivity. *IEEE/ACM transactions on networking* 29, 5 (2021), 2215–2227. https://doi.org/10.1109/tnet.2021.3084303

[33] Pierre-Joseph Proudhon and John Beverley Robinson. 2004. *General idea of the revolution in the nineteenth century*. Courier Corporation. Original work published 1851.

[34] Filippo Santoni de Sio and Jeroen van den Hoven. 2018. Meaningful Human Control over Autonomous Systems: A Philosophical Account. *Frontiers in Robotics and AI* 5 (2018), 15. https://doi.org/10.3389/frobt.2018.00015

[35] Gal Shahaf, Ehud Shapiro, and Nimrod Talmon. 2020. Genuine Personal Identifiers and Mutual Sureties for Sybil-Resilient Community Growth. In *International Conference on Social Informatics*. Springer, Springer, EU, 320–332. https://doi.org/10.1007/978-3-030-60975-7_24

[36] Ehud Shapiro. 2021. Multiagent Transition Systems for Composing Fault-Resilient Protocol Stacks. *arXiv preprint arXiv:2112.13650* (2021). https://arxiv.org/abs/2112.13650

[37] Ehud Shapiro. 2022. Grassroots Currencies: Turning Mutual Trust into Liquidity. *arXiv preprint arXiv:2202.05619* (2022). https://arxiv.org/abs/2202.05619

[38] Ehud Shapiro. 2023. Grassroots Distributed Systems: Concept, Examples, Implementation and Applications (Brief Announcement). In *37th International Symposium on Distributed Computing (DISC 2023). (Extended version: arXiv:2301.04391)*. LIPICS, Italy, 47:1–47:7. https://arxiv.org/abs/2301.04391

[39] Ehud Shapiro. 2023. Grassroots Social Networking: Serverless, Permissionless Protocols for Twitter/LinkedIn/WhatsApp. In *OASIS '23* (Rome, Italy). Association for Computing Machinery. https://doi.org/10.1145/3599696.3612898

[40] Ehud Shapiro. 2024. A Grassroots Architecture to Supplant Global Digital Platforms by a Global Digital Democracy. *arXiv:2404.13468, Proceedings of DAWO'24* (2024). https://arxiv.org/abs/2404.13468

[41] Ehud Shapiro. 2025. Characterising Global Platforms: Centralised, Decentralised, Federated, and Grassroots. *arXiv preprint arXiv:2511.03286* (2025). https://arxiv.org/abs/2511.03286

[42] Ehud Shapiro. 2025. GLP: A Grassroots, Multiagent, Concurrent, Logic Programming Language for AI (full version). *arXiv preprint arXiv:2510.15747, Summary in Proc. of ICLP'26, EPTCS 450, pp. 119–133* (2025). https://arxiv.org/abs/2510.15747

[43] Ehud Shapiro. 2026. Child-Safe Social Networking: From a Grassroots Social Contract to an App. *In preparation* (2026).

[44] Ehud Shapiro. 2026. Grassroots Bonds: Financing by the People, for the People. arXiv:2603.13671. arXiv:2603.13671 https://arxiv.org/abs/2603.13671

[45] Ehud Shapiro. 2026. Grassroots Platforms with Atomic Transactions: Social Graphs, Cryptocurrencies, and Democratic Federations. In *Proceedings of the 27th International Conference on Distributed Computing and Networking*. 71–81. https://doi.org/10.1145/3772290.3772309 arXiv preprint arXiv:2502.11299.

[46] Ehud Shapiro. 2026. Implementing Grassroots Logic Programs with Multiagent Transition Systems and AI (Full Version). *arXiv:2602.06934. Summary to appear in Proc. of LOPSTR+PPDP'26* (2026). https://arxiv.org/abs/2602.06934

[47] Ehud Shapiro. 2026. sGLP: Simulating People with Stochastic Agents. arXiv:2610.14738. arXiv:2610.14738 [cs.MA]

[48] Ehud Shapiro. 2026. Sovereign Grassroots Currencies: A CBDC Architecture for Credit and Monetary Policy. arXiv:2609.27727. arXiv:2609.27727 https://arxiv.org/abs/2609.27727

[49] Ehud Shapiro. 2026. Types for Grassroots Logic Programs (Full version). *arXiv preprint arXiv:2601.17957* (2026). https://arxiv.org/abs/2601.17957

[50] Ehud Shapiro and Nimrod Talmon. 2026. Constitutional Governance in Metric Spaces. *Proceedings of the International Conference on Algorithmic Decision Theory (ADT 2026), to appear* (2026). https://arxiv.org/abs/2605.13362 arXiv:2605.13362.

[51] Yonadav Shavit et al. 2023. *Practices for Governing Agentic AI Systems*. White paper. OpenAI. https://cdn.openai.com/papers/practices-for-governing-agentic-ai-systems.pdf

[52] Peter Slattery et al. 2026. The AI risk repository: A meta-review, database, and taxonomy of risks from artificial intelligence. *Patterns* (2026), 101517. https://doi.org/10.1016/j.patter.2026.101517 arXiv:2408.12622

[53] Nimrod Talmon and Ehud Shapiro. 2025. Grassroots Federation: Fair Democratic Governance at Scale. *arXiv preprint arXiv:2505.02208; also Proc. of AAMAS'26* (2025). https://arxiv.org/abs/2505.02208

[54] Dominic Tarr, Erick Lavoie, Aljoscha Meyer, and Christian Tschudin. 2019. Secure scuttlebutt: An identity-centric protocol for subjective and decentralized applications. In *Proceedings of the 6th ACM conference on information-centric networking*. 1–11. https://doi.org/10.1145/3357150.3357396

[55] Cristian Vaccari and Andrew Chadwick. 2020. Deepfakes and Disinformation: Exploring the Impact of Synthetic Political Video on Deception, Uncertainty, and Trust in News. *Social Media + Society* 6, 1 (2020). https://doi.org/10.1177/2056305120903408

[56] Laura Weidinger et al. 2022. Taxonomy of Risks Posed by Language Models. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*. Association for Computing Machinery, 214–229. https://doi.org/10.1145/3531146.3533088

[57] Gavin Wood et al. 2014. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper* 151, 2014 (2014), 1–32. https://ethereum.github.io/yellowpaper/paper.pdf

[58] Karen Yeung. 2018. Algorithmic regulation: A critical interrogation. *Regulation & Governance* 12, 4 (2018), 505–523. https://doi.org/10.1111/rego.12158

[59] Shoshana Zuboff. 2019. *The age of surveillance capitalism: The fight for a human future at the new frontier of power.* Public Affairs, US. https://www.hachettebookgroup.com/titles/shoshana-zuboff/the-age-of-surveillance-capitalism/9781610395694/

A The Framework, Recalled

This appendix recalls, in the words of [26], the mathematical foundations on which the paper is based; the proofs of the results recalled are in that paper.

Each agent consists of a *person* and a *machine* (e.g., smartphone) operated by the person. A *machine transaction* specifies what machines do; a *volition guard* specifies which persons must be willing for the transaction to occur.

We assume a potentially infinite set of *agents* Π, but consider only finite subsets of it, so we assume any particular set of agents $P \subset \Pi$ to be nonempty and finite. We use $\subset$ to denote the strict subset relation and $\subseteq$ when equality is also possible. As standard, we use S^P to denote the set of all total functions from P to S, and if $c \in S^P$ we use c_p to denote the value of c at $p \in P$.

Machine states, configurations, machine transactions, and volition-guarded transactions are as in Definition 3.1 (Section 3.2), which also recalls the reading of guards and of the unguarded case.

Distinct machine transactions can represent "the same action" in different configurations; this is captured with an equivalence relation on machine transactions.

Definition A.1 (Transaction Equivalence [26]). Given a set of machine transactions R, a *transaction equivalence* is an equivalence relation $\sim$ on R such that $t \sim t'$ implies t and t' have the same participants, and $R/\sim$ is countable. We write $[t]$ for the equivalence class of t under $\sim$.

For example, all befriend(p, q) transactions—differing only in the configurations in which they occur—form an equivalence class. Next is the mathematical machinery allowing people to express their volitions regarding the classes of machine transactions their machines may participate in.

Definition A.2 (Agent State and Configuration [26]). Given agents P, states S with initial state s_0, a set of machine transactions T each over its own participants $Q \subseteq P$ and S, and equivalence $\sim$ on T, an *agent state* is a pair $(V, m) \in \mathcal{A} = (2^{T/\sim} \times S)$ where V is its *volitional state* and $m \in S$ its *machine state*. The *initial agent state* is $(\emptyset, s_0)$. An *agent configuration* c over P, S, T, and $\sim$ is a member $c \in \mathcal{A}^P$ in which $c_p^v \subseteq (T/\sim)_p$ for every $p \in P$, where $(T/\sim)_p$ denotes the classes in $T/\sim$ in which p is a participant; we write c_p^v for the volitional state and c_p^m for the machine state of agent p in c.

Definition A.3 (Volitional Transaction [26]). Given agents P, states S, machine transactions T over P and S, and equivalence $\sim$ on T:

(1) A *volition change of agent* $p \in P$ is a pair $c \to c'$ of agent configurations over $\{p\}$, S, T, and $\sim$ such that $c_p^v, {c'}_p^v \subseteq (T/\sim)_p$ and $c_p^v \neq {c'}_p^v$, and $c_p^m = {c'}_p^m$.
(2) A *volitional machine transaction* induced by a volition-guarded transaction (t, Q'), for some $t = (d \to d') \in T$ over $Q' \subseteq Q \subseteq P$, is a pair $c \to c'$ where $c \neq c'$ are agent configurations over P, S, T, and $\sim$ such that $[t] \in c_q^v$ for every $q \in Q'$; $c_p^m = d_p$ and ${c'}_p^m = d'_p$ for every $p \in Q$; $c_p^m = {c'}_p^m$ for every $p \in P \setminus Q$; and ${c'}_p^v = c_p^v \setminus \{[t]\}$ for every $p \in P$.
(3) A *volitional transaction* is a volition change or a volitional machine transaction.

When a volitional machine transaction induced by (t, Q') is taken, the class $[t]$ is removed from every agent's volitional state: the will is fulfilled by any equivalent transaction. A person may independently change their volitional state via volition changes, which may add or remove classes; beyond these, the framework removes a class from c_p^v only upon fulfilment.

A person's not willing a transaction has, so far, no consequence: no volitional transaction is enabled by the absence of a will. An implementation of volition-guarded transactions by communicating agents [22] needs one, so that a machine asked for its person's will answers either way; it is the following, by which the machine answers for a person who does not will an act: in machine state s, if p wills it s' is taken, else s''.

Definition A.4 (Volition-Guarded If-Then-Else Transaction, Then-Branch, Else-Branch [26]). Given machine states S, a *volition-guarded if-then-else transaction* over a single participant $p \in \Pi$ is a triple $s \to s'$; s'' *guarded by* p, with $s, s', s'' \in S$, $s \neq s'$ and $s \neq s''$; the unary machine transaction $s \to s'$ is its *then-branch* and $s \to s''$ its *else-branch*.

Definition A.5 (Else Transaction [26]). Given agents P, states S, machine transactions T over P and S, and equivalence $\sim$ on T, an *else transaction* induced by a volition-guarded if-then-else transaction $s \to s'$; s'' guarded by $p \in P$, with $s \to s' \in T$, is a pair $c \to c'$ of agent configurations over P, S, T, and $\sim$ such that $[s \to s'] \notin c_p^v$; $c_p^m = s$ and ${c'}_p^m = s''$; $c_q^m = {c'}_q^m$ for every $q \in P \setminus \{p\}$; and ${c'}_q^v = c_q^v$ for every $q \in P$.

The then-branch, taken when p wills it, is the unary volition-guarded transaction $(s \to s', \{p\})$ and induces volitional machine transactions per Definition A.3; a unary volition-guarded transaction is thus the special case with no else-branch. The else transaction is taken when p does not will the then-branch; it fulfils no will and removes no class. The two are placed in one class below, so that liveness obliges the machine to answer either way; the obligation is the machine's, and it has no time bound. Only a unary transaction has an else-branch: a binary one would let an agent that never wills it trigger it.

Induced transition systems. The following is a simplified variation, sufficient for the purpose of this work, on the foundations introduced in [36].

Definition A.6 (Transition System, Computation, Run, Safe, Live, Correct [26]). A *transition system* is a tuple $TS = (S, s_0, T, \sim)$, where:

(1) S is an arbitrary non-empty set, referred to as the set of *states*.
(2) Some $s_0 \in S$ is the designated *initial state*.
(3) $T \subseteq S^2$ is a set of *correct transitions over* S, where each transition $t \in T$ is a pair (s, s') of non-identical states $s \neq s' \in S$, also written as $t = s \to s'$.
(4) $\sim$ is a *partial equivalence relation* on T: a symmetric and transitive relation on T, not necessarily reflexive. Its domain $\{t \in T : t \sim t\}$ is partitioned into countably many *liveness classes* $T/\sim$; a transition outside the domain belongs to no class.

A *computation* of TS is a (nonempty, potentially infinite) sequence of states $r = s_1, s_2, \cdots$; it is a *run* of TS if $s_1 = s_0$. A *prefix* of a computation is a finite initial segment of it. A computation $r =$

$s_1, s_2, \ldots$ is *safe*, also written $r \subseteq T$, if $s_i \to s_{i+1} \in T$ for every two consecutive states; and $s \xrightarrow{*} s' \subseteq T$ denotes the existence of a safe computation from s to s' (empty if $s = s'$). A class $[t] \in T/\sim$ is *enabled* in a state s if $s \to s' \in [t]$ for some $s' \in S$. A run r is *live* if no class $[t] \in T/\sim$ is enabled in every state of some suffix of r with no member of $[t]$ occurring in the suffix. A run is *correct* if it is safe and live.

A partial equivalence, rather than a total one, is used so that some transitions may carry no liveness requirement at all: only transitions in the domain of $\sim$ form classes and thereby incur a liveness obligation, while transitions outside the domain—belonging to no class—may occur in a correct run but are never required to. A run with uncountably many classes enabled in every state of a suffix is not live.

Lemma A.7 (Extension [26]). *Every finite safe run of a transition system is a prefix of a correct run of it.*

Definition A.8 (Multiagent Transition System [26]). Given agents $P \subset \Pi$ and an arbitrary set S of *states* with a designated *initial state* $s_0 \in S$, a *multiagent transition system* over P and S is a transition system $TS = (C, c_0, T, \sim)$ with *configurations* $C := S^P$, *initial configuration* $c_0 := \{s_0\}^P$, *transitions* $T \subseteq C^2$ a set of transactions over P and S, and $\sim$ a partial equivalence on T.

Rather than specifying a multiagent transition system over a set of agents P directly, it is specified via machine transactions (Definition 3.1). A machine transaction over $Q \subseteq P$ defines a set of multiagent transitions over P in which all members of $P \setminus Q$ are stationary:

Definition A.9 (Transaction Closure [26]). Let $P \subset \Pi$, S a set of machine states, and $C := S^P$. For any transition or transaction $t = c \to c'$, we write $t_q := c_q \to c'_q$ and say p is *stationary* in t if $c_p = c'_p$. For a machine transaction $t = (c \to c')$ over S with participants Q, the *P-closure of t*, $t{\uparrow}P$, is the set of transitions over P and S defined by:

$$t{\uparrow}P := \begin{cases} \{t' \in C^2 : \forall q \in Q.(t_q = t'_q) \wedge & \text{if } Q \subseteq P \\ \qquad \forall p \in P \setminus Q.(p \text{ is stationary in } t')\} & \\ \emptyset & \text{otherwise} \end{cases}$$

If R is a set of machine transactions, each $t \in R$ over some Q and S, then the *P-closure of R*, $R{\uparrow}P$, is the set of transitions over P and S defined by:

$$R{\uparrow}P := \bigcup_{t \in R} t{\uparrow}P$$

Given a relation $\sim$ on R, its *P-closure* $\sim{\uparrow}P$ is the relation on $R{\uparrow}P$ with $\hat{t}$ $(\sim{\uparrow}P)$ $\hat{t}'$ iff $\hat{t} \in t{\uparrow}P$ and $\hat{t}' \in t'{\uparrow}P$ for some $t \sim t'$. A transition over P is *unary* if it lies in the P-closure of a unary transaction.

Namely, the closure over $P \supseteq Q$ of a machine transaction t over Q includes all transitions t' over P in which members of Q do the same in t and in t', and the rest remain in their current (arbitrary) state. The closure likewise carries any relation on transactions to one on the induced transitions. If distinct transactions in R have disjoint P-closures—as when every participant of every transaction in R changes state—then each transition in $R{\uparrow}P$ has a unique inducing transaction, $\sim{\uparrow}P$ relates two transitions exactly when their inducing transactions are $\sim$-related, and $\sim{\uparrow}P$ is a partial equivalence whenever $\sim$ is.

A set R of volition-guarded transactions with an equivalence $\sim$ on their underlying machine transactions induces a volitional multiagent transition system:

Definition A.10 (Volitional Multiagent Transition System [26]). Given agents $P \subset \Pi$, machine states S with initial state s_0, a set R of volition-guarded transactions and volition-guarded if-then-else transactions—a volition-guarded if-then-else transaction $s \to s'\,;\, s''$ guarded by p standing in R for its then-branch $(s \to s', \{p\})$ together with its else-branch—such that every $(t, Q') \in R$ has the participants of t contained in P, every guard p of a volition-guarded if-then-else transaction is in P, and distinct underlying machine transactions, else-branches included, have disjoint P-closures, and an equivalence $\sim$ on the set $T_R := \{t : (t, Q') \in R \text{ for some } Q'\}$ of underlying machine transactions (a total equivalence, hence in particular a partial equivalence in the sense of Definition A.6), such that no else-branch is in T_R and volition-guarded if-then-else transactions with the same else-branch have $\sim$-equivalent then-branches, the *volitional multiagent transition system induced by* $(S, R, \sim)$ *over* P is the multiagent transition system $(\mathcal{A}^P, c_0, T_V, \sim_V)$ where:

(1) $\mathcal{A} := 2^{T_R/\sim} \times S$ is the *agent state space*;
(2) $c_0 \in \mathcal{A}^P$ is the *initial agent configuration*, with ${c_0}^v_p = \emptyset$ and ${c_0}^m_p = s_0$ for every $p \in P$;
(3) T_V consists of all transitions $e \to e' \in (\mathcal{A}^P)^2$ of one of three forms: (*i*) a *volition change* of some $p \in P$—$e^v_p, {e'}^v_p \subseteq (T_R/\sim)_p$ and $e^v_p \neq {e'}^v_p$, $e^m_p = {e'}^m_p$, and $e_r = e'_r$ for every $r \in P \setminus \{p\}$; (*ii*) a *volitional machine transaction* induced by some volition-guarded transaction $(t, Q') \in R$ per Definition A.3(2); or (*iii*) an *else transaction* induced by some volition-guarded if-then-else transaction in R per Definition A.5;
(4) $\sim_V$ is the restriction of $\sim{\uparrow}P$ (Definition A.9) to the volitional machine transactions and else transactions, relating two of them whenever their inducing machine transactions are $\sim$-equivalent, where the inducing machine transaction of an else transaction is the then-branch $s \to s'$ of its volition-guarded if-then-else transaction, and leaves every volition-change transition outside its domain—so volition changes belong to no class and carry no liveness obligation.

The conditions on else-branches ensure that the machine transactions inducing a transition of T_V are $\sim$-equivalent, so that $\sim_V$ is a partial equivalence. The set R of volition-guarded transactions that specifies a platform thus determines the volitional machine transitions of the induced volitional multiagent transition system, while volition-change transitions are freely available to every agent independent of R.

Definition A.11 (Enabled [26]). Given a set of volition-guarded transactions, each (t, Q') with $t = d \to d'$ a machine transaction over some $Q' \subseteq Q \subseteq P$ and S, and an equivalence $\sim$ on machine transactions: the volition-guarded transaction (t, Q') is *machine-enabled* in agent configuration c over P if $c^m_p = d_p$ for every $p \in Q$, and *enabled* if in addition $[t] \in c^v_q$ for every $q \in Q'$. The else-branch of a volition-guarded if-then-else transaction $s \to s'\,;\, s''$ guarded by p is *enabled* in c if $c^m_p = s$ and $[s \to s'] \notin c^v_p$. An equivalence class $[t]$ is *enabled* in c if some volition-guarded transaction (t', Q') with $t' \in [t]$ is enabled in c, or the else-branch of some volition-guarded if-then-else transaction with then-branch $t' \in [t]$ is enabled in c.

A volition-guarded transaction with an empty guard ($Q' = \emptyset$) requires no volitions and is enabled whenever its machine precondition is met. The class of the then-branch of a volition-guarded if-then-else transaction guarded by p is thus enabled whenever p's machine is in its source state, whether or not p wills it. For the volitional multiagent transition system above, a class of volitional machine transactions and else transactions is enabled at a configuration in the sense of Definition A.6 exactly when some volition-guarded transaction or else-branch inducing it is enabled in the above sense; this determines which runs are live and correct per Definition A.6. Volition-change transitions belong to no class and so impose no liveness obligation; personal choices remain free.

Protocols and grassroots protocols. A protocol is a family of multiagent transition systems, one for each set of agents $P \subset \Pi$, which share an underlying set of local machine states $\mathcal{S}$ with a designated initial state s_0. A *local-states function* maps every set of agents $P \subset \Pi$ to an arbitrary set of local machine states $S(P) \subset \mathcal{S}$ that includes s_0 and satisfies $P \subset P' \subset \Pi \implies S(P) \subset S(P')$.

Definition A.12 (Protocol [26]). A *protocol* $\mathcal{F}$ over a local-states function S is a family of multiagent transition systems that has exactly one transition system $\mathcal{F}(P) = (C(P), c_0(P), T(P), \sim(P))$ for every $P \subset \Pi$, with *agent states* $\mathcal{A}(P)$, configurations $C(P) := \mathcal{A}(P)^P$, initial configuration $c_0(P) \in C(P)$, and partial equivalence $\sim(P)$ on $T(P)$ determined by the protocol, such that $P \subseteq P' \subset \Pi$ implies $\mathcal{A}(P) \subseteq \mathcal{A}(P')$ and $c_0(P)_p = c_0(P')_p$ for every $p \in P$.

Definition A.13 (Interleaving [26]). Let $P, P' \subset \Pi$ be disjoint nonempty sets of agents, $r = c_0, c_1, \ldots$ a run of $\mathcal{F}(P)$, and $r' = d_0, d_1, \ldots$ a run of $\mathcal{F}(P')$. An *interleaving* of r and r' is a sequence $e_0, e_1, \ldots$ of configurations in $C(P \cup P')$ for which there exist nondecreasing sequences of indices $(i_k)_{k\geq 0}$ and $(j_k)_{k\geq 0}$ with $i_0 = j_0 = 0$ such that for every $k \geq 0$:

(1) $(e_k)_p = (c_{i_k})_p$ for every $p \in P$,
(2) $(e_k)_q = (d_{j_k})_q$ for every $q \in P'$,
(3) if e_{k+1} exists, then exactly one of: (*i*) $i_{k+1} = i_k + 1$ and $j_{k+1} = j_k$ (a P-step), or (*ii*) $i_{k+1} = i_k$ and $j_{k+1} = j_k + 1$ (a P'-step).

Moreover, if r is finite of length n then $i_k = n$ for some k, and if r is infinite then for every $m \geq 0$ there is a k with $i_k = m$; likewise for r' and (j_k).

Definition A.14 (Interaction, Interactive Run, First Interaction [26]). Let $Q \subset \Pi$. For a configuration c over Q and $R \subseteq Q$, write $c|_R$ for the restriction of c to R. A transition $e \to e'$ of $\mathcal{F}(Q)$ is an *interaction between* $x \neq y \in Q$ if $e|_{\{x\}} \neq e'|_{\{x\}}$ and $e|_{\{y\}} \neq e'|_{\{y\}}$, and $e|_{\{x\}} \to e'|_{\{x\}}$ is not a transition of $\mathcal{F}(\{x\})$ or $e|_{\{y\}} \to e'|_{\{y\}}$ is not a transition of $\mathcal{F}(\{y\})$. It is an *interaction between* disjoint nonempty $P, P' \subseteq Q$ if it is an interaction between some $x \in P$ and some $y \in P'$. A run $\hat{r}$ of $\mathcal{F}(Q)$ is *interactive between P and P'* if some transition of $\hat{r}$ is an interaction between them; the *first interaction* of P and P' in $\hat{r}$ is the first such transition.

Definition A.15 (Oblivious, Interactive, Grassroots [26]). A protocol $\mathcal{F}$ is:

(1) *oblivious* if for every disjoint nonempty $P, P' \subset \Pi$, every interleaving of a correct run of $\mathcal{F}(P)$ and a correct run of $\mathcal{F}(P')$ is a correct run of $\mathcal{F}(P \cup P')$.
(2) *interactive* if for every $Q \subset \Pi$ and disjoint nonempty $P, P' \subseteq Q$, every prefix of a correct run of $\mathcal{F}(Q)$ containing no interaction between P and P' is a prefix of a correct run of $\mathcal{F}(Q)$ that contains one.
(3) *grassroots* if it is oblivious and interactive.

Being oblivious means two disjoint groups coexist without interference: any interleaving of a correct run of one with a correct run of the other is a correct run of the combined system. Being interactive means two groups that have not yet interacted can still do so: the combined system has a correct continuation in which a single transaction engages agents of both groups. Agents that have taken part in a common interaction are joined; an instance is a maximal set of agents joined directly or indirectly, and two instances coalesce when they come to be one.

Definition A.16 (Interaction Graph, Instance, Coalescence [26]). Let $\mathcal{F}$ be a protocol, $Q \subset \Pi$, and r a prefix of a run of $\mathcal{F}(Q)$. The *interaction graph* of r has vertices Q and an edge between $a \neq b \in Q$ whenever some transition of r is an interaction between a and b. A nonempty $P \subseteq Q$ is an *instance* in r if it is a connected component of the interaction graph of r. Disjoint nonempty $P, P' \subseteq Q$ *coalesce* in a run $\hat{r}$ of $\mathcal{F}(Q)$ if they are instances in some prefix of $\hat{r}$ and lie in one component of the interaction graph of a longer prefix of $\hat{r}$.

Definition A.17 (Transactions Over a Local-State Function [26]). Let S be a local-states function. A set of transactions R is *over S* if every transaction $t \in R$ is a multiagent transition over Q and $S(P')$ for some $Q \subseteq P' \subset \Pi$. Given such a set R and $P \subset \Pi$, $R(P) := \{t \in R : t \text{ is over } Q \text{ and } S(P'), Q \subseteq P' \subseteq P\}$.

Definition A.18 (Volitional Transactions-Based Protocol [26]). Let S be a local-states function and R a set of volition-guarded transactions over S with equivalence $\sim$. The *protocol $\mathcal{F}$ over R, S, and $\sim$* assigns to each set of agents $P \subset \Pi$ the volitional multiagent transition system $\mathcal{F}(P)$ induced by $(S(P), R(P), \sim)$ over P (Definition A.10). In particular, $C(P) = \mathcal{A}(P)^P$ with agent state space $\mathcal{A}(P) := 2^{T_{R(P)}/\sim} \times S(P)$—monotone in P, as Definition A.12 requires—and $c_0(P)$ has p-component $(\emptyset, s_0)$ for every $p \in P$.

The obliviousness results rest on the following invariant: in a group's own system an agent wills only transactions internal to the group, so the guard of a transaction that reaches another group cannot will it while the group runs alone, and the transaction is never enabled in an interleaving.

Lemma A.19 (Volitional Containment [26]). *Let $\mathcal{F}$ be a volitional transactions-based protocol over a set of volition-guarded transactions R with equivalence $\sim$. For every $P \subset \Pi$, every configuration c of $\mathcal{F}(P)$, and every $p \in P$: $c_p^v \subseteq T_{R(P)}/\sim$, where $T_{R(P)} := \{t : (t, Q') \in R(P) \text{ for some } Q'\}$.*

Lemma A.20 (Interleaving Safety [26]). *Let $\mathcal{F}$ be a volitional transactions-based protocol and $P, P' \subset \Pi$ disjoint and nonempty. Every finite prefix of an interleaving of a correct run of $\mathcal{F}(P)$ and a correct run of $\mathcal{F}(P')$ is a finite safe run of $\mathcal{F}(P \cup P')$.*

Proposition A.21 ([26]). *A volitional transactions-based protocol is oblivious provided that for every disjoint nonempty $P, P' \subset \Pi$, no equivalence class whose transactions have participants spanning both P and P' is ever enabled in any interleaving of correct runs of $\mathcal{F}(P)$ and $\mathcal{F}(P')$.*

Corollary A.22 (Guarded Obliviousness [26]). *A volitional transactions-based protocol over a set of volition-guarded transactions R is oblivious if every volition-guarded transaction $(t, Q') \in R$ whose*

machine transaction t has two or more participants has a nonempty guard, $Q' \neq \emptyset$.

In a run of $\mathcal{F}(P \cup P')$ a member may will a transaction reaching the other group, which a group's own run forbids; the volitional notion constrains the resulting coupling:

Definition A.23 (Left [26]). Let $\mathcal{F}$ be a volitional transactions-based protocol over R, $Q \subset \Pi$, c a configuration of $\mathcal{F}(Q)$, and $x \in Q$. Then x *has left* in c if no $(t, Q') \in R(Q)$ with $x \in Q'$ is machine-enabled in any configuration c' of $\mathcal{F}(Q)$ with $c \xrightarrow{*} c' \subseteq T(Q)$.

Definition A.24 (Interactive Among the Agents That Have Not Left [26]). A volitional transactions-based protocol $\mathcal{F}$ is *interactive among the agents that have not left* if for every $Q \subset \Pi$ and disjoint nonempty $P, P' \subseteq Q$, every prefix of a correct run of $\mathcal{F}(Q)$ containing no interaction between P and P', in whose last configuration some agent of P and some agent of P' have not left, is a prefix of a correct run of $\mathcal{F}(Q)$ that contains one.

An interactive protocol is interactive among the agents that have not left, the latter requiring a continuation of fewer prefixes. A group all of whose agents have left can never be connected by an act one of them wills, and the definition asks nothing of it.

Definition A.25 (Volitionally Grassroots [26]). A volitional transactions-based protocol $\mathcal{F}$ is *volitionally grassroots* if it is oblivious, interactive among the agents that have not left, and, for every disjoint nonempty $P, P' \subset \Pi$ and every safe run of $\mathcal{F}(P \cup P')$ interactive between P and P', the first interaction of P and P' is induced by a volition-guarded transaction (t, Q') with $Q' \cap P \neq \emptyset$ and $Q' \cap P' \neq \emptyset$.

Two groups thus become connected only by an act a person of each wills, and a grassroots protocol meeting the condition on first interactions is volitionally grassroots. An else transaction (Definition A.5) is over a single participant and so is never an interaction; a binary transaction with an else-branch would let an agent of another group, who never wills it, trigger it, and could be a first interaction willed by neither, which is why Definition A.4 admits only unary ones.

B The Transactions of the Contract

Fifteen transactions, written in the local-state components of Section 3.2, one per speech-act schema of the contract (`programs/gai/GAI.glp` and `programs/gai/GSNP.glp` in the GLP repository, their records `cascade/GAI` and `cascade/GSNP`), each determined by its schema at each binding by the compilation of [17], and carrying the schema's name. Every transaction requires $\iota = 1$ at each of its participants, except Establish, and Enter at the AI agent, and $\theta = 0$ at each of its guards; the requirements are left out below, and all components not explicitly updated are preserved.

(3) Volition-Guarded Multiagent Atomic Transactions

(1) **Establish**(p), where $\iota_p = 0$: $\iota'_p := 1$, guarded by p.
(2) **Acquaint**(p, q), where $p \neq q$ and $a_p = a_q = 0$: $\nu'_p := \nu_p \uplus \{q\}$ and $\nu'_q := \nu_q \uplus \{p\}$, guarded by $\{p, q\}$.
(3) **Introduce**$(p, q; c)$, where $a_p = a_q = 0$, $q \in \nu_p$, $c \in \nu_p$ and $p \in \nu_q$, for an agent c: $\nu'_q := \nu_q \uplus \{c\}$, guarded by $\{p, q\}$.
(4) **Befriend**(p, q), where $p \neq q$, $a_p = a_q = 0$, $q \in \nu_p$, $p \in \nu_q$ and $q \notin c_p$: $c'_p := c_p \cup \{q\}$ and $c'_q := c_q \cup \{p\}$, guarded by $\{p, q\}$.
(5) **Unfriend**(p, q), where $q \in c_p$ and $q \notin \rho_p \cup \kappa_p$: $c'_p := c_p \setminus \{q\}$ and $c'_q := c_q \setminus \{p\}$, guarded by p.
(6) **Enter**$(p, r; k)$, where $a_p = 0$, $k \in h_p$ and $\iota_r = 0$: $\kappa'_p := \kappa_p \cup \{r\}$, $c'_p := c_p \cup \{r\}$, $\nu'_p := \nu_p \uplus \{r\}$, $\epsilon'_p := \epsilon_p \cup \{(r, p, sgn(k, p))\}$, $\iota'_r := 1$, $a'_r := 1$, $\rho'_r := \{p\}$, $c'_r := \{p\}$, $\nu'_r := \{p\}$ and $\gamma'_r := \{sgn(k, p)\}$, guarded by $\{p, r\}$.
(7) **Agent acquaint**$(r, q, p; y)$, where $a_r = 1$, $\rho_r = \{p\}$, $y \in \gamma_r$, $(r, p, y) \in \epsilon_p$ and $a_q = 0$: $\nu'_r := \nu_r \uplus \{q\}$, $\nu'_q := \nu_q \uplus \{r\}$ and $\epsilon'_q := \epsilon_q \cup \{(r, p, y)\}$, guarded by $\{r, q, p\}$.
(8) **Agents acquaint**$(r, s, p, q; y, z)$, where $r \neq s$, $a_r = a_s = 1$, $\rho_r = \{p\}$, $\rho_s = \{q\}$, $y \in \gamma_r$, $z \in \gamma_s$, $(r, p, y) \in \epsilon_p$ and $(s, q, z) \in \epsilon_q$: $\nu'_r := \nu_r \uplus \{s\}$, $\epsilon'_r := \epsilon_r \cup \{(s, q, z)\}$, $\nu'_s := \nu_s \uplus \{r\}$ and $\epsilon'_s := \epsilon_s \cup \{(r, p, y)\}$, guarded by $\{r, s, p, q\}$.
(9) **Agent befriend**(r, q, p), where $a_r = 1$, $\rho_r = \{p\}$, $q \in \nu_r$, $a_q = 0$, $r \in \nu_q$ and $q \notin c_r$: $c'_r := c_r \cup \{q\}$ and $c'_q := c_q \cup \{r\}$, guarded by $\{r, q, p\}$.
(10) **Agents befriend**(r, s, p, q), where $r \neq s$, $a_r = a_s = 1$, $\rho_r = \{p\}$, $\rho_s = \{q\}$, $s \in \nu_r$, $r \in \nu_s$ and $s \notin c_r$: $c'_r := c_r \cup \{s\}$ and $c'_s := c_s \cup \{r\}$, guarded by $\{r, s, p, q\}$.
(11) **Agent unfriend**(r, q, p), where $\rho_r = \{p\}$, $q \in c_r$ and $q \neq p$: $c'_r := c_r \setminus \{q\}$ and $c'_q := c_q \setminus \{r\}$, guarded by p.
(12) **Retire**(p, r), where $a_r = 1$, $\rho_r = \{p\}$, $\theta_r = 0$ and $p \in c_r$: $c'_p := c_p \setminus \{r\}$, $\tau'_p := \tau_p \cup \{r\}$, $c'_r := c_r \setminus \{p\}$ and $\theta'_r := 1$, guarded by p.
(13) **Sign**$(p; x)$, where x is a speech act signed by p: $h'_p := h_p \uplus \{x\}$ and $\alpha'_p := \alpha_p \cup \{x\}$, guarded by p.
(14) **Message**$(p, q; x)$, where $x \in \alpha_p$ and $q \in c_p$: $h'_q := h_q \uplus \{x\}$, guarded by p.
(15) **Forward**$(p, q; x)$, where $x \in h_p$, $x \notin \alpha_p$ and $q \in c_p$: $h'_q := h_q \uplus \{sgn(x, p)\}$, guarded by p, where $sgn(x, p)$ is x signed over by p.

The schema `unfriend` is guarded in its first role, the friend ending the friendship, and is bound either way, so Unfriend(p, q) is guarded by p and Unfriend(q, p) by q: either friend may end the friendship, unless one is an AI agent of the other, a friendship that only Retire ends (GnW:3, GnW:4 and GAI:4). The requirement $q \notin \rho_p \cup \kappa_p$ of Unfriend stands for the atoms *principal*(q) and *principal_of*(q) forbidden at p, and $q \neq p$ of Agent unfriend for *principal*(q) forbidden at r. The requirement $\rho_r = \{p\}$ of Agent acquaint, Agents acquaint, Agent befriend, Agents befriend, Agent unfriend and Retire stands for the two atoms *principal*(p) at r and *principal_of*(r) at p, equivalent in every run, as Enter sets both together and no transaction changes either; the schemas require both. The requirement $k \in h_p$ of Enter stands for *item*(k) at p, and $y \in \gamma_r$ of Agent acquaint and Agents acquaint for *attestation*(y) at r, with $(r, p, y) \in \epsilon_p$ for its record *agent_of*(r, p, y) at p, which Enter adds and from which the principal's machine fills the request; $x \in \alpha_p$ of Message stands for *authored*(x) at p, and $x \notin \alpha_p$ of Forward for *authored*(x) forbidden there. The requirement $\theta = 0$ at each guard stands for *retired* forbidden at every guarding role. The component ν_p is a multiset: Acquaint, Agent acquaint and Agents acquaint neither require nor forbid that the two know each other,

and add a copy of each at the other whether they do or not, so that two agents who know each other may take the act again.

Transaction equivalence. The equivalence $\sim$ is that of [17]: one schema at one admissible binding determines one class, whose members differ only in the configuration in which they occur. All Acquaint(p, q) transactions form one class, and so do all Introduce$(p, q; c)$, all Befriend(p, q), all Unfriend(p, q), all Agent befriend(r, q, p), all Agents befriend(r, s, p, q), all Agent unfriend(r, q, p) and all Retire(p, r) with the same agents; all Enter$(p, r; k)$, all Agent acquaint$(r, q, p; y)$ and all Agents acquaint$(r, s, p, q; y, z)$ with the same agents and speech acts; each Establish(p); each Sign$(p; x)$ with the same item; and each Message$(p, q; x)$ and each Forward$(p, q; x)$ with the same agents and item.

C Contracts with Wards

This appendix defines contracts with wards, agents who act only with the consent of a guardian, as an extension of the speech-act schema language of [17]: a child with their parents [43], an AI agent with its principal. The GnW contract (Section 3.1) marks a ward as one and lets only the unsupervised come to know each other (GKK:2), so a contract extending it has no act that any two agents can take and is not syntactically grassroots in the sense of [17]; the protocol realising it is volitionally grassroots nonetheless, since a ward enters the contract with a guardian recorded, who is not a ward, and comes to know, through their guardian, anyone their guardian can come to know, and a retired ward has left. Section C.1 recalls the language and the results of [17] this appendix uses, with the two additions the extension makes to the language, reflexive predicates and signed terms; Section C.2 states conditions on the schemas under which a contract with wards is certified, decidable as the conditions of [17] are, and proves them sufficient (Theorem C.20); Section C.3 certifies the social network with provenance of Section 4 (Proposition C.21) and the contract of this paper (Proposition C.22). The checker of [17], in the repository https://github.com/EShapiro2/GLP, decides the conditions of this appendix as well.

C.1 The Language and the Results Recalled

Atoms and states. Π is a countable, potentially infinite set of *agents* and Δ a countable set of *utterances* disjoint from it. A *speech act* is an utterance or a speech act signed by an agent: D is the least set with $\Delta \times \Pi \subseteq D$ and $D \times \Pi \subseteq D$, the *signer* of $(x, p) \in D$ is p, and $D_P := \{d \in D : \text{signer}(d) \in P\}$ for $P \subseteq \Pi$ [6, 18]. Σ is a set of *predicates*, each with an arity; an *atom* is a predicate applied to as many arguments as its arity, a syntactic object with no truth value, and it *names* what is among its arguments or is the signer of one. A *local state* is a finite multiset of atoms over $\Pi \cup D$, the initial state s_0 is empty, and $S_C(P)$ is the set of finite multisets of atoms over $P \cup D_P$ [17]. Multiset sum, difference, intersection and inclusion are $\uplus$, $\setminus$, $\cap$ and $\subseteq$, and the *join* $\bigsqcup_j s_j$ takes the greatest multiplicity. A *mark* is a predicate of arity zero, and its atom is written as the mark. A schema is written over disjoint sets R of *roles*, X of *agent variables* and Y of *speech-act variables*, disjoint from Π and D.

Definition C.1 (Speech-act schema [17]). A *speech-act schema* is written $\tau(\pi_1, \ldots, \pi_k; x_1, \ldots, x_m) : E_1 \cdots E_k$, with τ its name, $\pi_1, \ldots, \pi_k \in R$ distinct, each possibly carrying the mark ? and those that do its *guarding roles*, $k \geq 1$ its *arity*, $x_1, \ldots, x_m \in X \cup Y$ distinct, its *variables*, T_τ the set of its *signed terms* $\text{sgn}(y, \pi)$ with y one of its variables in Y and π one of its roles, and each E_i a finite multiset of atoms over $\{\pi_1, \ldots, \pi_k, x_1, \ldots, x_m\} \cup T_\tau$, each carrying at most one of the marks $-$, $+$, $\neg$. For each role π_i the marks divide E_i into $+(i)$, $-(i)$, $\neg(i)$ and $=(i)$, its members marked $+$, $-$, $\neg$ and unmarked; the atoms π_i *requires* are $=(i) \uplus -(i)$, and it *forbids*, *adds* and *deletes* those of $\neg(i)$, $+(i)$ and $-(i)$. It is required that $+(i) \cap -(i) = \emptyset$, that $+(i) \uplus -(i) \neq \emptyset$ at every role that is not a guarding role and at some role, that a signed term occurs only in an atom marked $+$, and that its speech-act variable occurs in an atom its role requires. A *contract* C is a finite set of speech-act schemas over Σ.

Definition C.2 (Binding [17]). A *binding* β for a schema τ maps its roles and agent variables to Π and Y to D, with $\beta(\text{sgn}(y, \pi)) := (\beta(y), \beta(\pi))$, distinct roles to distinct agents, but for two the schema marks as possibly one, which it may send to one agent if neither of them adds or deletes an atom naming the other, and is extended to atoms and to multisets of them componentwise. It is *admissible* if every speech-act variable of τ that occurs in an atom added at a role and in no atom required at a role is mapped to a speech act signed by the agent in a guarding role of τ.

Reflexive predicates. Two roles sent to one agent are carried out by that agent together. A role that adds or deletes an atom naming another cannot share its agent, so no relation a contract records is reflexive. The extension lets a contract declare a predicate of arity one *reflexive*: every state is then taken to hold the atom of that predicate naming its holder, and no schema adds or deletes, at a role, the atom of it naming that role. For a configuration d and an agent p, $\hat{d}_p$ is d_p together with the atom naming p of every reflexive predicate; in a contract declaring none, $\hat{d}_p = d_p$. The child-safe social network declares *friend* reflexive, so that two children with one parent may befriend by an act in which that parent takes both guardian roles [43]; the contract of this paper declares no reflexive predicate.

Signed terms. The extension lets a schema add at a role an atom with a *signed term* $\text{sgn}(y, \pi)$ among its arguments, the speech act y signed by the agent in the role π: a signed term occurs only in an atom marked $+$, its speech-act variable occurs in an atom its role requires, a binding β sends it to $(\beta(y), \beta(\pi))$, and in traceable provenance it counts as the two arguments y and π (Definitions C.1, C.2 and C.6). Forward adds at the recipient *item*$(\text{sgn}(x, \textit{Alice}))$, what Alice held, signed by Alice, and Enter adds at the AI agent *attestation*$(\text{sgn}(k, p))$, the code signed over by the principal (Appendix B). In [17] a speech-act variable is bound to a speech act signed by an agent in a guarding role, and no schema signs over what a role holds.

Definition C.3 (Compilation [17]). Let τ be a schema with roles $\pi_1, \ldots, \pi_k$ and β a binding for it, and for an agent p let $I_p := \{i : \beta(\pi_i) = p\}$. A machine transaction $d \to d'$ over $\{\beta(\pi_1), \ldots, \beta(\pi_k)\}$ is *determined by* τ *at* β if for every i, with $p := \beta(\pi_i)$, $\beta(=(i) \uplus -(i)) \subseteq \hat{d}_p$ and $\beta(\neg(i)) \cap \hat{d}_p = \emptyset$, and for every agent p that β sends a role to, $\bigsqcup_{i \in I_p} \beta(+(i)) \cap \bigsqcup_{i \in I_p} \beta(-(i)) = \emptyset$ and $d'_p = (d_p \setminus \bigsqcup_{i \in I_p} \beta(-(i))) \uplus \bigsqcup_{i \in I_p} \beta(+(i))$; the volition-guarded transaction it determines is the pair of it with $\{\beta(\pi) : \pi \text{ a guarding role of } \tau\}$.

Table 1: The local state of an agent p.

	component	stands for
identity, knowing and friends	$\iota_p \in \{0,1\}$, whether p has a genuine digital identity	*identity*
	$a_p \in \{0,1\}$, whether p is an AI agent	*agent*
	ν_p, a multiset of agents, those p knows, each by their public key	*knows*
	$c_p \subseteq P$, the friends of p	*friend*
guardianship	$\rho_p \subseteq P$, the principal of p, at most one, and $\kappa_p \subseteq P$, the AI agents of p	*principal*, *principal_of*
	γ_p, the attestation of p, a set of at most one speech act	*attestation*
	ϵ_p, triples (r, q, y) of an AI agent r, its principal q and its attestation y that p holds recorded	*agent_of*
	$\theta_p \in \{0,1\}$, whether p is retired, and $\tau_p \subseteq P$, the AI agents p retired	*retired*, *retired_of*
items	h_p, a multiset of speech acts, the items p holds, each signed by its author and by every agent that forwarded it, in order	*item*
	$\alpha_p \subseteq h_p$, the items p authored	*authored*

Table 2: The components that change in the example.

Transactions	Alice	Bob	Alf
Establish(a), Establish(b)	$\iota_a = 1$	$\iota_b = 1$	
Acquaint(a, b)	$\nu_a = \{b\}$	$\nu_b = \{a\}$	
Befriend(a, b)	$c_a = \{b\}$	$c_b = \{a\}$	
Sign$(a; k)$	$h_a = \alpha_a = \{k\}$		
Enter$(a, r; k)$	$\kappa_a = \{r\}$, $\nu_a = \{b, r\}$, $c_a = \{b, r\}$		$\iota_r = 1$, $a_r = 1$, $\rho_r = \{a\}$, $\nu_r = c_r = \{a\}$, $\gamma_r = \{y\}$
Agent acquaint$(r, b, a; y)$		$\nu_b = \{a, r\}$, $\epsilon_b = \{(r, a, y)\}$	$\nu_r = \{a, b\}$
Agent befriend(r, b, a)		$c_b = \{a, r\}$	$c_r = \{a, b\}$
Sign$(r; x)$, Message$(r, b; x)$		$h_b = \{x\}$	$h_r = \alpha_r = \{x\}$
Retire(a, r)	$\tau_a = \{r\}$, $c_a = \{b\}$		$\theta_r = 1$, $c_r = \{b\}$
Unfriend(b, r)		$c_b = \{a\}$	$c_r = \emptyset$

The *realisation* R_C of C is the set of volition-guarded transactions determined by its schemas at all admissible bindings, $\sim$ the equivalence on their machine transactions generated by relating two when one schema at one admissible binding determines both, and the *protocol realising* C the volitional transactions-based protocol over R_C, S_C and $\sim$ [17, 26]; R_C with $\sim$ is well formed (Lemma The compilation is well defined of [17]), whose proof, like the compilation, uses nothing of the introductory act, so both stand for the contracts of this appendix.

The protocol. From [26], recalled in Appendix A: for a set R of volition-guarded transactions over S and $P \subset \Pi$, $R(P)$ holds those of them over $S(P')$ for some $P' \subseteq P$, and R is *contained* if every transaction of it with participants in P and source over $S(P)$ is in $R(P)$; the protocol $\mathcal{F}$ over R, S and $\sim$ has at each $Q \subset \Pi$ the transition system $\mathcal{F}(Q)$ whose transitions are volition changes and the volitional machine transactions the members of $R(Q)$ induce, each *machine-enabled* in a configuration whose machine states restrict on its participants to its source, and *enabled* if every guard holds its class besides; a run is *safe* if its consecutive configurations form transitions, and *correct* if besides no class stays enabled forever with no member taken. Every finite safe run is a prefix of a correct run (Lemma Extension of [26]).

Definition C.4 (Records; recording graphs [17]). A local state *records* an agent q if one of its atoms names q; a machine state s records q if $s \notin S(P)$ for every $P \subset \Pi$ with $q \notin P$; on local states the two agree (Lemma Records of [17]). The *recording graph* of a configuration d is the directed graph on Π with an edge $p \to q$ whenever d_p records q, and for a set E of predicates its *E-recording graph* is the subgraph whose edges $p \to q$ are those for which d_p holds an atom with predicate in E naming q; the recording graph of a run is the union of those of its configurations. A volition-guarded transaction $(d \to d', \{p, q\})$ over participants $\{p, q\}$ is a *mutual recording of p and q* if it changes the state of each and the recording graph of d' has the edges $p \to q$ and $q \to p$.

Definition C.5 (Introductory act, unobstructed [17]). A schema τ of C of arity two, guarded in both roles, is an *introductory act* if any two agents $p \neq q$ have an *introduction*: an admissible binding β of τ into $\{p, q\}$ that sends π_1 to p and π_2 to q, under which some atom added at π_1 names q and some atom added at π_2 names p. It is *unobstructed* if any two agents $p \neq q$ have an introduction β such that, at each role π_i of τ:

(1) every atom τ requires at π_i, other than an atom of a reflexive predicate naming π_i, is, under β, added by some schema of C of arity one that is guarded in its role, requires nothing, and forbids nothing or only that atom, which τ then requires once at π_i, at some admissible binding that sends its role to $\beta(\pi_i)$;
(2) every schema of C that, at some binding, adds at $\beta(\pi_i)$ an atom τ forbids at π_i has, at that binding, the other of p and q in a role and adds at $\beta(\pi_i)$ an atom naming them;
(3) τ forbids at π_i no atom of a reflexive predicate naming π_i.

The exceptions for reflexive predicates in clauses 1 and 3 are the extension's; without reflexive predicates the definition is that of [17].

Definition C.6 (Traceable provenance, volition [17]). Write Φ_E for the atoms over $R \cup X \cup Y$ and signed terms whose predicate is in E; a signed term $\mathrm{sgn}(y, \pi)$ among the arguments of an atom counts as the two arguments y and π. A set $E \subseteq \Sigma$ has *traceable provenance* in C if for every schema τ of C with roles $\pi_1, \ldots, \pi_k$, every i, every $\varphi \in +(i) \cap \Phi_E$ and every argument u of φ in $R \cup X$ with $u \neq \pi_i$, u is a role of τ or an argument of some $\psi \in (=(j) \uplus -(j)) \cap \Phi_E$, and every argument of φ in Y is an argument of such a ψ or occurs in no atom required at a role of τ; a predicate has traceable provenance if it belongs to such a set, and C has it if the set of its predicates has it. The *role graph* of a schema is the undirected graph on its roles with an edge between π_i and π_j when one of the two requires an atom of traceable provenance with the other among its arguments, and C satisfies *volition* if every schema of it has a connected role graph or a guarding role in every connected component of its role graph.

C is *syntactically grassroots* if it has an unobstructed introductory act, traceable provenance and volition.

Definition C.7 (Open, closed, polite [17]). A set R of volition-guarded transactions is *open* if for any two agents $p \neq q$ in any $Q \subset \Pi$, every finite safe run of $\mathcal{F}(Q)$ with no interaction between p and q has a finite safe extension, by volition changes and transitions that are not interactions between p and q, at whose end some transaction of $R(Q)$ whose induced transitions are interactions between p and q is machine-enabled; *closed* if for every $Q \subset \Pi$, every transaction of R enabled at the end of a finite safe run of $\mathcal{F}(Q)$ has all its participants in one instance in that run or a guard in every instance in that run holding one of them; and *polite* if it is open and closed.

The results of [17] used below, with C a contract of that paper's language, $\mathcal{F}$ the protocol realising it, $Q \subset \Pi$ and r a finite safe run of $\mathcal{F}(Q)$.

Lemma C.8 ([17]). *(1)* Every participant changes state. *Every transition of $\mathcal{F}(Q)$ induced by a transaction of R_C changes the state of every participant of it.*
(2) A mutual recording is an interaction. *Every transition of $\mathcal{F}(Q)$ induced by a mutual recording of p and q in $R(Q)$ is an interaction between p and q.*
(3) Containment. *If C has traceable provenance, R_C is contained.*
(4) No edge crosses instances. *If E has traceable provenance in C, every edge of the E-recording graph of r joins two agents that lie in one instance in r.*
(5) Closure. *If C has traceable provenance and volition, R_C is closed.*
(6) Openness. *If C has an unobstructed introductory act, R_C is open.*
(7) Containment and closure. *The protocol over a contained, closed set of volition-guarded transactions is oblivious, and the first interaction of two disjoint sets of agents in a safe run of it is induced by a transaction guarded by a member of each.*
(8) Polite. *The protocol over a contained, polite set of volition-guarded transactions is volitionally grassroots.*

Parts 1, 2 and 7 are proved in [17] for every set of volition-guarded transactions, part 7 being the parts of the proof of its Theorem Polite headed Oblivious and Volitionally grassroots, which use containment and closure and not openness; parts 3 to 6 are proved there for a contract without reflexive predicates, and the next lemma extends them.

Lemma C.9 (Reflexive predicates and signed terms). *Parts 3 to 6 of Lemma C.8 hold for a contract declaring reflexive predicates and adding atoms with signed terms, with Definition C.5 in place of the definition of unobstructed of [17].*

Proof. Each proof of [17] goes through with one case added, where an atom a role requires is, under the binding, the atom of a reflexive predicate naming the agent in that role, which $\hat{d}$ holds at every agent. Containment: an argument of an atom added, by traceable provenance an argument of an atom ψ required at a role π_j, is $\beta(\pi_j)$ itself when $\beta(\psi)$ is that atom, an agent of Q. No edge crosses instances: for an argument s, or a speech-act variable y, of the added atom that is an argument of an atom ψ required at a role j, with $p' := \beta(j)$, $\beta(\psi)$ is the atom of a reflexive predicate naming p', and then $q = p'$, or an atom of the state of p', as there. Closure: the image of the atom of traceable provenance that π_i requires naming π_j lies in the state of $p := \beta(\pi_i)$ or is the atom of a reflexive predicate naming p, and then $q = p$ and the two lie in one instance. Openness: the extension of the run adds, at each role, every atom required there other than an atom of a reflexive predicate naming its holder, which $\hat{d}$ holds; by clause 3 of Definition C.5 no such atom is forbidden; and no schema adds or deletes at a role the atom of a reflexive predicate naming it, so the rest of the proof is unchanged. A signed term occurs only in an added atom, so openness and closure are untouched by it; Containment: an argument of an added atom that is $(\beta(y), \beta(\pi))$ for a signed term is signed by $\beta(\pi)$, an agent of Q, so the state stays in $S_C(P)$; No edge crosses instances: where $q = \beta(s)$ for the role s of a signed term among the added atom's arguments, q is a participant, joined to p by an edge of the interaction graph. □

C.2 Guardianship, Leaving and Relay

Throughout, M is a set of marks of C.

Definition C.10 (Guardianship). A *guardianship* for M in C is (m, ι, g, w, v), with $m \in M$ and ι marks and g, w and v predicates of arity one, v possibly w, such that:
(1) no schema of C deletes ι, and no schema adds or deletes a mark in M at more than one role, or two different marks in M at one role;
(2) a schema that adds m at a role adds ι there and forbids ι there;
(3) a schema adds, and deletes, m at a role as many times as it adds, and deletes, atoms of g there;
(4) an atom $g(u)$ a schema adds at a role ρ has u a role $\sigma \neq \rho$ of the schema at which the schema requires ι, forbids every mark in M and adds $w(\rho)$; and a schema deletes $g(u)$ at ρ if and only if it deletes $w(\rho)$ at the role u;
(5) a schema that adds $w(\rho)$ at a role adds $v(\rho)$ there; a schema that adds $v(\rho)$ at a role requires ι there and forbids every mark in M there; and a schema that deletes $v(\rho)$ at a role deletes or forbids $w(\rho)$ there.

The ward x holds $g(a)$ of its creator a, who holds $w(x)$; every guardian of x, the creator among them, holds $v(x)$ (GnW:1). For AI agents the marks are *agent* and *identity*, the predicates *principal* and *principal_of*, and v is *principal_of*, the principal being the AI agent's one guardian: Enter adds *agent*, *identity* and *principal*(p) at the AI agent, who had no identity, and *principal_of*(r) at the principal p, who has one and is not an AI agent (Appendix B). For children they are *child* and *identity*, the predicates *creator* and *created*, and v is *parenting*, which every parent holds [43].

Lemma C.11 (Wards and guardians). *Let C have a guardianship for every mark in M and (m, ι, g, w, v) be one of them. In every configuration of a run of the protocol realising C, an agent holds m as many times as they hold atoms of g; an agent holding $g(a)$ holds ι, while a holds w of them and ι and no mark in M; an agent holding $w(x)$ holds $v(x)$; and an agent holding $v(x)$ holds ι and no mark in M.*

Proof. By induction on the run. The initial state is empty. A transition is induced by a transaction determined by a schema at a binding, and changes a state only by the atoms the schema adds and deletes there. Clause 3 keeps the counts of m and of g-atoms equal at every role, and by clause 1 at every agent, as a schema adds

or deletes them at one role. If the transition adds $g(a)$ at p then by clause 3 it adds m at p, by clause 2 it adds ι at p, and by clause 4 its schema has a at a role requiring ι and forbidding every mark in M and adds $w(p)$ at a: so after it p holds ι, and a holds $w(p)$, ι and no mark in M. These persist: ι is never deleted (clause 1); a mark in M is added at a role only where ι is forbidden (clause 2 of its guardianship), so an agent holding ι never gains one; and $w(p)$ is deleted at a only when $g(a)$ is deleted at p (clause 4). By clause 5, a transition adding $w(x)$ at an agent adds $v(x)$ there, and one deleting $v(x)$ there deletes $w(x)$ or finds it absent, so an agent holding $w(x)$ holds $v(x)$; and a transition adding $v(x)$ at an agent finds ι there and no mark in M, which persist as above. □

Definition C.12 (Leaving mark). Let C have a guardianship for every mark in M. A mark $\ell \notin M$ is a *leaving mark* of C if:

(1) every guarding role of every schema of C forbids ℓ, and no schema of C deletes ℓ;
(2) a schema that adds ℓ at a role requires a mark in M there;
(3) a schema that adds, at a role, an atom $g(\sigma)$ of the guardianship of a mark in M forbids ℓ at σ.

Retirement under GnW:4 is written with a leaving mark: the act by which the creator retires a ward adds it at the ward; here it is *retired*, added by Retire.

Lemma C.13 (Leaving). *Let C have a guardianship for every mark in M and ℓ be a leaving mark of C. In every configuration of a run of the protocol realising C, an agent holding ℓ has left; and if an agent holds $g(a)$ for the guardianship (m, ι, g, w, v) of a mark in M, then a does not hold ℓ.*

Proof. No schema deletes ℓ (clause 1), so an agent holding ℓ holds it in every later configuration. A transaction (t, Q') with that agent in Q' sends them a guarding role, which forbids ℓ (clause 1), so no such transaction is machine-enabled in a later configuration, and the agent has left. When an atom $g(a)$ is added at an agent, a does not hold ℓ (clause 3), and the act adds none at a, whose role forbids every mark in M (Definition C.10, clause 4; clause 2); and while an agent holds $g(a)$, a holds no mark in M (Lemma C.11), so no transition adds ℓ at a (clause 2). □

Definition C.14 (Unobstructed but for marks). An introductory act τ of C that adds one atom at each role, of a predicate e of arity one, is *unobstructed but for M* if at each role π_i of τ:

(1) every atom τ requires at π_i, other than an atom of a reflexive predicate naming π_i, is, up to renaming the role, the one atom added by some schema of C of arity one that is guarded in its role, requires nothing and forbids nothing but that atom and the leaving marks of C;
(2) every atom τ forbids at π_i is a mark in M or a leaving mark of C.

The act neither requires nor forbids an atom of e, so two agents may take it whether or not each holds e of the other, and a local state, a multiset, counts the copies; an agent may hold e of an agent they never dealt with, as GKK:3 introduces one agent to another. Acquaint, which forbids *agent* at each role and adds *knows* of the other there, is unobstructed but for $\{agent\}$: it requires *identity*, which Establish adds (Appendix B).

Definition C.15 (Relay act). Let C have a guardianship for every mark in M and τ be an introductory act of C unobstructed but for M, adding atoms of e. A schema χ of C is a *relay act* with *related roles* $\rho_1 \neq \rho_2$, roles of it guarded in χ at each of which χ adds an atom of e naming the other, if:

(1) each related role ρ is a *ward role* of some $m \in M$, requiring m, with exactly one other role σ of χ, its *guardian role*, requiring $v(\rho)$, the v of the guardianship of m, a guardian of the ward (GnW:2), or a *free role*, forbidding every mark in M; every role of χ is a related role or a guardian role; and χ adds and deletes at a guardian role no atom naming another guardian role;
(2) every atom χ requires at a role is an atom of a reflexive predicate naming the role; or, up to renaming the role, the one atom of a schema of arity one as in clause 1 of Definition C.14; or, at a ward role, its m or $g(\sigma)$; or, at a ward role of m, an atom of a predicate h whose arguments are distinct speech-act variables of χ occurring in no other atom χ requires but its record below, where every schema of C that adds m at a role adds there an atom of h whose arguments are speech-act variables or signed terms and adds, at the role its g-atom names, its *record*, an atom of a predicate h' whose arguments are the two roles and the arguments of the h-atom, and no schema of C deletes an atom of h or of h'; or, at the guardian role σ of a ward role ρ, the atom $v(\rho)$ or $w(\rho)$ of that guardianship, or the record, the atom of h' whose arguments are ρ, σ and the arguments of the h-atom ρ requires; or, at a guardian role or a free role, $e(\kappa)$ for another guardian role or free role κ, with e reflexive where both are guardian roles;
(3) every atom χ forbids at a role is a mark in M or a leaving mark of C.

A relay act is an act by which a ward and another agent come to know each other, with the guardian of each ward among them. Agent acquaint is a relay act with a ward role, the AI agent, a free role, the person, and the principal as the guardian role, requiring at the AI agent the attestation its principal gave it on entering, an atom of h in clause 2, and at the principal its record, from which the principal's machine fills the request; Agents acquaint is one with two ward roles and their two guardian roles. In the child-safe social network a child and an adult come to know each other by a relay act with the child's parent as the guardian role, requiring of the parent *knows* of the adult, and two children by one with two ward roles and their two parents, requiring of each parent *knows* of the other, which binds both parents to one person where the two children share one, *knows* being reflexive [43]. Befriending, which requires *knows* of the other at each role, is not a relay act in either contract.

Definition C.16 (Introduction with wards). C has an *introduction with wards* if it has a nonempty set M of marks, a guardianship for each mark in M, an introductory act unobstructed but for M, and, for every $m, m' \in M$, a relay act with a ward role of m and a free role and a relay act with a ward role of m and a ward role of m'.

Throughout the rest of this appendix C is a contract with an introduction with wards, traceable provenance and volition, R_C its realisation and $\mathcal{F}$ the protocol realising it.

Definition C.17 (Open among the agents that have not left). A set R of volition-guarded transactions is *open among the agents that*

have not left if for any two agents $p \neq q$ in any $Q \subset \Pi$, every finite safe run of $\mathcal{F}(Q)$ with no interaction between p and q, in whose last configuration neither p nor q has left, has a finite safe extension, by volition changes and transitions that are not interactions between p and q, at whose end some transaction of $R(Q)$ whose induced transitions are interactions between p and q is machine-enabled.

An open set is open among the agents that have not left.

Proposition C.18 (Openness with wards). *R_C is open among the agents that have not left, and open if C has no leaving mark.*

Proof. Let $p \neq q$ in Q, let r be a finite safe run of $\mathcal{F}(Q)$ with no interaction between p and q, and let c be its last configuration, at which, if C has a leaving mark, neither p nor q has left; by Lemma C.13, neither holds a leaving mark at c. Call $x \in Q$ a *ward* at c if its state holds a mark in M, which by Lemma C.11 is one mark m and comes with $g(a)$ for some a, its creator, whom we take as its *guardian*, who holds $w(x)$, $v(x)$ and ι and no mark; otherwise x is *free*. Two facts. First, the transition that added $g(a)$ at a ward x was determined by a schema with a in a role at which it added $w(x)$ (Definition C.10, clause 4), so it changed both states (Lemma C.8(1)), left x recording a, and was an interaction between x and a; so the guardian of p, if p is a ward, is not q, and that of q is not p. Second, for free $x \neq y$ in Q holding no leaving mark, the introductory act τ at the introduction of x and y can be made machine-enabled by unary transitions at x and at y: an atom τ requires that is absent is the one atom of a schema of arity one requiring nothing and forbidding nothing but that atom and the leaving marks, which is machine-enabled, and adding it adds nothing τ forbids, as a schema of arity one requiring nothing adds no mark in M, such a mark being added only together with a g-atom naming another role (Definition C.10, clauses 3 and 4), and no leaving mark, which is added only where a mark in M is required (Definition C.12, clause 2); and what τ forbids is a mark in M, absent at the free x and y, or a leaving mark, absent at x and y. Taking τ there, after volition changes of x and y, leaves x holding $e(y)$ and y holding $e(x)$, and is an interaction between x and y and no others.

If p and q are free, the second fact at $x := p$ and $y := q$ gives an extension by unary transitions after which τ at their introduction is machine-enabled, and every transaction it determines is a mutual recording of p and q, so its transitions are interactions between p and q by Lemma C.8(2).

Otherwise let χ be the relay act for the statuses of p and q, bound with its ward roles to those of p and q that are wards, its free role to the free one, and the guardian role of each ward role to that ward's guardian, and the speech-act variables of each atom of a predicate h that χ requires at a ward role, as clause 2 of Definition C.15 admits, to the arguments of an atom of h held by the ward bound there, which holds one: the transition that added its mark added one, and no transition deletes one; the same transition added the record, the atom of h', at that ward's guardian, and no transition deletes one, so the guardian holds it under the binding. The related roles are bound to distinct agents, and two guardian roles are bound to one person only where p and q share a guardian, which GnW:2 admits, Alice and Dave possibly one person, and Definition C.2 permits as the guardian roles add and delete nothing naming each other. Extend r as follows, by transitions none of which is an interaction between p and q: for every atom χ requires at a role that is the one atom of a schema of arity one and is absent at the agent bound there, the unary transition adding it, machine-enabled as above; and for every $e(\kappa)$ required at a guardian role or free role bound to x, with κ bound to $y \neq x$, where x does not hold $e(y)$, the unary transitions and the transaction of τ at the introduction of x and y by the second fact, x and y being free by Lemma C.11, holding no leaving mark, as p and q hold none and a guardian holds none by Lemma C.13, and $\{x, y\} \neq \{p, q\}$, since a guardian is bound only where its ward is one of p and q, and is not the other by the first fact. At the end χ is machine-enabled: a reflexive atom naming its holder is in $\hat{d}$; the unary atoms are present; a ward role bound to p has m, $g(a)$ and the atom of h its variables are bound to at p; a guardian role bound to a has w and v of its ward, by Lemma C.11, and the record of its ward's h-atom at a; $e(\kappa)$ at a role bound to x with κ bound to y holds by the extension where $y \neq x$ and by reflexivity where $y = x$; a mark in M that χ forbids is absent at the free agents and at the guardians, by Lemma C.11, and a leaving mark is absent at p, at q and at the guardians, by Lemma C.13. Every transaction χ determines there leaves p holding an atom naming q and q one naming p, changing both states, so each records the other after it, the transition restricted to p is no transition of $\mathcal{F}(\{p\})$, and it is an interaction between p and q. □

Theorem C.19 (Polite among the agents that have not left). *The protocol over a contained set of volition-guarded transactions that is closed and open among the agents that have not left is volitionally grassroots.*

Proof. Let R be such a set and $\mathcal{F}$ the protocol over it. By Lemma C.8(7) $\mathcal{F}$ is oblivious, and the first interaction of two disjoint sets of agents in a safe run is induced by a transaction guarded by a member of each. Let $Q \subset \Pi$, let $P, P' \subseteq Q$ be disjoint and nonempty, and let $\hat{r}$ be a prefix of a correct run of $\mathcal{F}(Q)$ containing no interaction between P and P', in whose last configuration $p \in P$ and $q \in P'$ have not left. By Definition C.17 some finite safe extension of $\hat{r}$, by volition changes and transitions that are not interactions between p and q, ends where some $(t, Q') \in R(Q)$ whose induced transitions are interactions between p and q is machine-enabled. If a transition of the extension is an interaction between P and P', the extension contains one. Otherwise extend it by a volition change of each agent of Q', where needed, so that $[t]$ lies in the volitional state of each, and by the volitional machine transaction that (t, Q') then induces, an interaction between p and q and hence between P and P'. Either way the result is a finite safe run of $\mathcal{F}(Q)$ having $\hat{r}$ as a prefix and containing an interaction between P and P', and by Lemma Extension of [26] it is a prefix of a correct run of $\mathcal{F}(Q)$. So $\mathcal{F}$ is interactive among the agents that have not left, and volitionally grassroots. □

Theorem C.20 (Grassroots realisation with wards). *The protocol realising a contract with an introduction with wards, traceable provenance and volition is volitionally grassroots.*

Proof. By Proposition C.18 R_C is open among the agents that have not left. The proof of closure (Lemma C.8(5)) uses traceable provenance and volition and nothing of the introductory act, so R_C is closed, and by Lemma C.8(3), whose proof uses traceable provenance only, it is contained; both hold with reflexive predicates

by Lemma C.9. By Theorem C.19 the protocol over R_C is volitionally grassroots. □

Where C has no leaving mark, R_C is open by Proposition C.18, hence polite, and by Lemma C.8(8) the protocol realising C is grassroots as well. A contract with wards whose groups are closed by a listed role of their members, as the child-safe social network's are, is certified by composing this appendix with the agent-named contracts of [17], in [43].

C.3 The Contracts Certified

The social network with provenance (Section 4) is written as speech-act schemas in the repository, `programs/gai/GSNP.glp`: the schemas of the contract of Section 3 (Appendix B), with message and forward; and the schemas of a group, which name a group by its leader and a number and close it by a listed role, the leader with every member the leader lists, as the agent-named contracts of [17]; create and join forbid the mark of an AI agent, so an AI agent takes no group act, the ward's membership of GnW:2 being left out (Section 3.2).

Proposition C.21 (The social network with provenance is grassroots). *The protocol realising GSNP is volitionally grassroots.*

Proof. The contract holds the schemas of the contract of Section 3 with `message`, `forward` and the schemas of a group. It has an introduction with wards exactly as in the proof of Proposition C.22: the marks $M = \{agent\}$, the guardianship (*agent*, *identity*, *principal*, *principal_of*, *principal_of*), the leaving mark *retired*, forbidden at every guarding role, the group schemas' among them, `acquaint` the introductory act unobstructed but for M, and `agent_acquaint` and `agents_acquaint` the relay acts. The set of all the predicates has traceable provenance: the schemas of Section 3's contract as in the proof of Proposition C.22; message adds $item(x)$ at Bob, whose speech-act variable the required $authored(x)$ carries; forward adds $item(\text{sgn}(x, Alice))$ at Bob, whose arguments are the role Alice of the signed term and the speech-act variable x, which the required $item(x)$ carries; and of the schemas of a group, create adds atoms whose arguments are its role, the name based on it and a speech-act variable it requires nowhere, join, close, post, post_own, post_forward, deliver and deliver_own add atoms whose arguments are roles, the roles of signed terms, that name and speech-act variables an agent role of the schema requires, and leave and drop add nothing. Volition holds: the schemas of Section 3's contract as in the proof of Proposition C.22; the role Alice of message and of forward requires $friend(Bob)$, an atom of traceable provenance naming the other role; join is guarded in both its roles, the two roles of close have the base Alice, create and post_own have arity one, and the role Alice of leave, drop, deliver and deliver_own requires $listed(Bob, n)$ and the role Bob of post and post_forward requires $member(\langle Alice\rangle \cdot n)$, each an atom of traceable provenance an argument of which has the base of the other role. Every name term is based on the role Alice and the listed role of close adds no atom, so the contract is agent-named in the sense of [17], with signed terms by Lemma C.9, and it has an introduction with wards; the protocol realising it is volitionally grassroots by composing Theorem C.20 with that paper's Theorem Grassroots realisation with listed roles, the composition in [43]. □

Proposition C.22 (Grassroots). *The protocol realising the contract (Appendix B) is volitionally grassroots.*

The contract is not syntactically grassroots: Acquaint forbids *agent*, which Enter adds at the AI agent by an act of the AI agent and its principal, so for an AI agent and an agent other than its principal clause 2 of Definition C.5 fails, and Befriend requires *knows* of the other, which no schema of arity one adds; the contract has no unobstructed introductory act. It has an introduction with wards, with the AI agents as its wards and *retired* as a leaving mark, forbidden at every guard, and the proof is by Theorem C.20.

Proof. The contract has an introduction with wards (Definition C.16), traceable provenance and volition (Definition C.6), so the protocol realising it is volitionally grassroots by Theorem C.20. The marks are $M = \{agent\}$, and (*agent*, *identity*, *principal*, *principal_of*, *principal_of*) is a guardianship for *agent* (Definition C.10): no schema deletes *identity*, *agent*, *principal* or *principal_of*, and `enter`, the one schema adding *agent*, adds it with *identity* and $principal(p)$ at the AI agent, where it forbids *identity*, and adds $principal_of(r)$ at p, where it requires *identity* and forbids *agent*; v is w, so clause 5 holds. The mark *retired* is a leaving mark (Definition C.12): every guarding role forbids it, no schema deletes it, `retire` adds it at a role requiring *agent*, and `enter` forbids it at p. `acquaint` is an introductory act unobstructed but for M (Definition C.14): it requires *identity*, the one atom of `establish`, which requires nothing and forbids nothing but the identity it adds and *retired*; and it forbids *agent* and *retired*. `agent_acquaint` is a relay act (Definition C.15) with the AI agent's role a ward role, the person's role a free role and the principal's role its guardian role, and `agents_acquaint` one with two ward roles and their two guardian roles; neither adds or deletes anything at a guardian role, and each forbids only *agent* and *retired*. At a ward role each requires *identity*, *agent*, *principal* of its guardian role and $attestation(y)$, and at the guardian role its record *agent_of*, the variable y occurring in no other atom, as clause 2 of Definition C.15 admits: `enter`, the one schema adding *agent*, adds the atom of *attestation* at the AI agent and its record at the principal, and no schema deletes either. Every atom a schema adds names only roles of that schema, speech acts a role of it requires or no role of it requires, and, in `introduce`, an agent variable of an atom required at a role: the signed term $sgn(x, p)$ that `enter` adds at the AI agent counts as its two arguments, the item x, which p requires, and the role of p, as in GSNP's `forward` [17]; `agent_acquaint` and `agents_acquaint` add $agent_of(r, p, y)$ with $attestation(y)$ required at r; and `sign` adds $item(x)$ and $authored(x)$, x required at no role: the contract has traceable provenance. In every schema the roles are joined by required atoms naming one another, or a guard stands in each part—`enter`, whose two roles share no record, is guarded in both, and the person's role of `agent_acquaint`, which requires nothing naming the other two, is guarded: the contract satisfies volition. □