From the philosophy of personal identity to the laws of agent societies

Ladislau Bölöni
School of Electrical and Computer Engineering
University of Central Florida
Orlando, Florida, 32816
Email: lboloni@cpe.ucf.edu

Abstract

Techniques frequently encountered in software agents, such as mobility, adaptation, cloning, splitting and merging put the identity of the participating agents in question. This in its turn, leads to problems concerning the responsibilities and rights of agents. Problems of personal identity are explored in the philosophical and psychological literature. This paper explores the applicability of this research for the case of software agents. We show how the concepts of somatic and psychological identity can lead to the development of a rule set for naming conventions in agent societies.

I. Introduction

A number of techniques used when building or deploying agents are affecting the identity of agents, raising more difficult philosophical and practical problems. Agent mobility implies transporting the code and state information of the of the agent over the network, and terminating the agent on the source host. One of the potential ways in which an agent migration can fail is by *accidental cloning*. Agents can be explicitly cloned [1], for reasons such as load-balancing or fault tolerance. In a more complex scenario, agents can be split into several parts, potentially having duplicated subcomponents. In the opposite operation, agents can be merged together. Typically, the merged agents are the results of a previous split operation, but other scenarios are also possible. In the case of user interface agents, such as a website, a user can have a conversational interaction with an agent which claims to be the same one with which the interaction happened a month ago. Users can accept this claim even if both the hardware and the software behind the agent was updated, and the company dissolved in a corporate takeover[1]. Agents can perform dynamic adaptation and mutation, which can sometimes radically change the behavior of the agent.

All these situations bring up the question of the *identity* of agents. This question is related to the purely technical questions of naming and addressing, but also raises problems of philosophical, psychological and legal nature. We argue that the implementation problems can not be solved in a consistent and satisfactory way without investigating the deeper philosophical issues involved.

This article is organized in two main sections. In section II we investigate the philosophical and psychological background of the identity in agents, comparing them with the similar problems involving the identity of humans. In section III we apply our conclusions towards practical problems of naming and addressing software agents for cases when the identity of the agents are in doubt. We conclude in section IV.

II. Philosophical foundations of identity. Humans vs. agents

The problem of personal identity was extensively researched in philosophy, psychology, law and religious studies. Is the philosophically identifiable person responsible for its actions if he was ”seized by

[1]This is a frequent situation. A online brokerage website is a direct replacement for a human broker. In the interaction with a human broker, the interaction is based on the assumption of the maintained identity of the participants.

the devil"? Or, in a more modern formulation, if he was brainwashed by an extremist group? Or, in case of a software agent, if it was modified by a malicious computer virus?

The problem of personal identity can be decomposed into a number of subproblems, which in the Stanford Encyclopedia of Philosophy [2] are identified as follows: (a) Who am I? The self-definition of a person. (b) The persistence question: is a particular person identical to a person in a particular moment of the past? (c) The evidence question: what is sufficient to prove the identity of two persons (for example: fingerprints). (d) The population question: how many persons are in a human body? (e) Personhood: what is sufficient and necessary for some entity to be considered a person? (f) What matters? What is the practical importance of the facts about the identity and persistence?

Problems (b) and (c) have immediate and direct applications in software agents. Problems (d) and (f), although not applicable directly, are also relevant in some circumstances. The problem (e) has an interesting parallel in the extensive disputes of early agent researchers about what attributes a software application needs have to be considered an agent [3]. Problem (a), at the current level of sophistication of software agents, is not relevant to agent research.

In the following we discuss the subproblems (b),(c) and (d) in greater detail. Our approach is to state the problems as discussed in philosophical studies, restate them for software agents, point out the similarities and differences, and finally, investigate how much of the philosophical results can be transferred in the software agent practice.

A. Persistance of identity

The usual way to state the persistence of identity question is: *Under what possible circumstances is a person existing at one time identical with a person existing at another time?*

The *Psychological Approach* assumes that a psychological relation between the two entities is necessary and/or sufficient to consider the two persons identical. A simplistic version of this approach is the *Memory Criterion*, sometimes attributed to Locke [4]: a person is identical with a person in the past if it can remember the person's past experiences. Unfortunately, this answer assumes a person with perfect memory - imperfect memories lead either to an unintuitive answer, or an identity relationship which is not transitive. A different approach proposed by Shoemaker [5] is based on causal dependence: two persons are considered identical if they are connected by a continuous causal chain of psychological connections.

The *Somatic Approach* ties the identity of the person to the physical object, the human animal which embodies it. The continuity of the persons are based on brute physical continuity.

A third view, sometimes called the *Simple View* denies that these or other identifiers of identity are necessary and sufficient. In this view, psychologic or physical continuity are just probabilistic signs of the personal identity. The identity is a self-containing definition, it can not be decomposed or proved with a set of simpler concepts. In fact, the proponents of this view point out that there is no need for continuity of the person for the survival of the human animal.

One of the important issues related to the persistency problem is the problem of *fission*. In human terms, this is usually stated with a thought experiment of the brain of a person being replicated and embedded in the other person. This lead to a famous debate between Parfit and Lewis [6].

The persistence question can be immediately applied to software agents, by simply replacing the term "person" with "agent". All three classes of answers have more or less immediate counterparts.

The somatic answer can be applied directly in the case of *embodied agents* such as robots, or agents which are permanently tied to a physical device[2]. One of the great appeals of the somatic approach is that it is easily decidable and its answer corresponds to the common sense in all practical situations concerning human persons or embodied agents.

For purely software agents, the somatic answer can be still applied, by associating the agents with the software artifact to which they are tied. For instance, we can associate an agent with the particular

[2]For example, software agents in the read-only memory of a PDA.

Unix process. Unfortunately, this gives an answer contrary to our intuition in many cases: agents migrated from a machine to another, or agents hibernated and restarted would not maintain their identity under this definition. We will call this definition *somatic identity* and note it with $\stackrel{s}{=}$.

The psychological answer is mapable to the case of agents with the following formulation: an agent is identical to an another agent in the past if its internal state (knowledgebase, model of the world, etc) can be traced back with an uninterrupted series of transformations to the state of the previous agent. These transformations are triggered by internal and external events, which, in case of humans would be called *experiences*. We call the relation defined this way *psychological identity* and note it with $\stackrel{p}{=}$. The memory criterion, for example, can be applied virtually unchanged to the case of software agents. Unfortunately it suffers from the same drawbacks: the memory of agents is not perfect. Furthermore, agents can selectively discard memories, and memories can be seemlessly inplanted in agents.

It is easy to see that for software agents, if two agents have the same identity according to the somatic criteria, they will have the same identity according to the psychological criteria as well: $A \stackrel{s}{=} B \Rightarrow A \stackrel{p}{=} B$. This follows from our the deterministic nature of software processes: their states can be always logically traced back to the state of the process at a previous time.

The opposite relation, however, is not true. A migrated agent, for example, maintains its identity according to the psychological criteria, but not according to the somatic criteria. Agents created by cloning or splitting have the same identity according the psychological criteria.

We note that the psychological and somatic identities are more divergent in case of agents then for humans. For humans, the two definitions differ only in the extreme cases of existence: for foetuses, humans in a permanent vegetative state or a number of hypothetical though experiments proposed by philosophers. At this point, we need to make the observation, that a number of thought experiments used in the philosophical study of identity are frequent events in a software agent's lifecycle. The phenomena of fission is usually studied through the though experiment of a brain transplant. In the world of software agents, this corresponds to a simple `fork()` call.

There are two major cases when the psychological criteria maintains that two agents have the same identity, while the somatic criteria denies it.

- $A(t) \stackrel{p}{=} B(t+\Delta t)$, $A(t) \stackrel{s}{\neq} B(t+\Delta t)$ and $\forall C, C(t+\Delta t) \stackrel{p}{=} B(t+\Delta t) \Rightarrow C(t+\Delta t) \stackrel{s}{=} B(t+\Delta t)$, i.e. B is the only agent at time $t+\Delta t$ which is p-identical to A at time t. This is the case of the migrated, or hibernated and restarted agents. We call B a *continuation* of the agent A.
- $A(t) \stackrel{p}{=} B(t+\Delta t)$ and $\exists C C(t+\Delta t) \stackrel{p}{=} B(t+\Delta t)$ and $C(t+\Delta t) \stackrel{s}{\neq} B(t+\Delta t)$. We call the agents B and C *siblings* if they aware of each others' existence. We call them *clones* if they are not aware of each others existence, i.e. both of them considers itself the continuation of agent A.

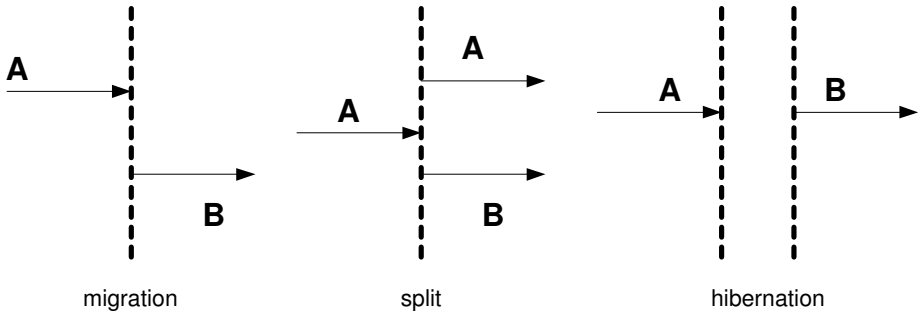

Fig. 1. The evolution of the somatic identity of an agent for migration, split and hibernation operations. All the agents in the picture are psychologically identical.

The case of hibernated agents is somewhat equivalent to humans in coma. The rights and responsibilities

of humans in such a state are temporarily interrupted. For example, in the US governmental structure, the vice-president temporarily assumes the responsibilities of the president, if the president is incapacitated. The fact that we do not treat this as a hiatus in the continuity of the personal identity is a purely verbal distinction. However, the somatic identity is maintained for humans in coma, while it is not for the hibernated software agents.

An example for siblings are agents which are split or replicated for the purpose of load balancing. These agents might be *identical*, but their *identity* is not under question. On the other hand, agents duplicated as a result of an interrupted migration are *clones*: neither of the agents is aware of the existence of the other agent. The agent in the destination site sees itself as the continuation of the original agent after a succesful migration, while the agent on the source site is the considers that he had just attempted an unsuccesful migration. An external entity might duplicate agents for malicious purposes, such as bank fraud.

In philosophical theories regarding fission if an agent has a sibling or clone, then it does *not* have the same identity as an agent in the past, independently of what the psychological or somatic criteria would say. On the other hand, if at a certain moment $t + \Delta t + \Delta t_2$ agent B becomes the only one which satisfies the psychological identity criteria with A, then it will become a continuation of the agent A. This situation can appear either as a result of agent C terminating, or as a result of a merge between agents B and C (see Figure 2).

This definition implies that the identity might have hiatuses in time. This creates major problems in the practice of software agents[3]. One way to get around this difficulty is to require that one of the siblings to be always designated as a continuation of the ancestor agent A. This leads however, to further difficulties about how the continuation is chosen (especially, if the agents are clones).

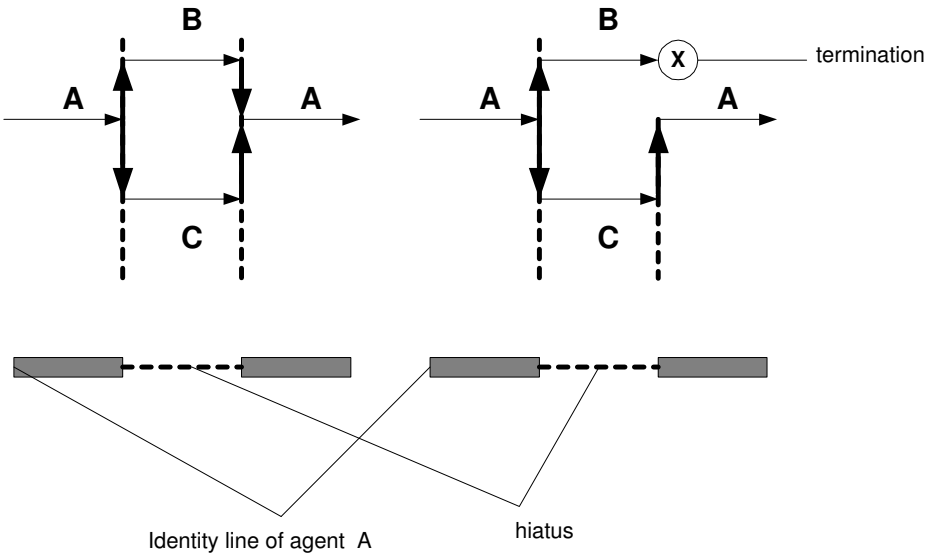

Fig. 2. The evolution of the continuation of an agent after a fission according to the prevailing philosophical theory. There can be hiatuses in the agent's lifeline.

[3]It would create problems in the case of humans as well. This situation however, is purely theoretical for the case of humans, while a frequent situation for software agents.

B. The evidence of identity. Introspection vs. external view

The philosophical study of the personal identity is motivated by introspection; ”Who am I?”. The problem of self-identifying has a much shallower meaning for software agents[4], but this does not make the involved problems trivial. The agent has a set of goals and intentions in its knowledgebase. Should he pursue these goals? Or are these the goals of an agent from which he was accidentally cloned or split? An external entity claims that it received a set of commitments from the agent: should the agent fulfill these commitments [7]–[9]?

Agents are interacting with humans and can serve as placeholders for one. It is likely that in the near future we will see software agents involved in legal procedures - both as a defendant and as a plaintiff. In these situations the positive identification of the agents involved will be necessary, and the identification might need to convince a jury.

In conclusion, we need to consider both the view of the agent about its own identity, and the perception of an external observer. Regarding the latter, there are two important aspects: the agent proving its own identity (authentication) and the agent accepting responsibility for its actions (nonrepudiation). We are generally interested in finding out if a particular agent is the *continuation* of a particular agent in the past.

One approach to proving identity is generally relying on some type of special knowledge of the agent, for example, knowledge of a password or a private key. These approaches are extensively studied in the computer security literature [10], but usually the actors involved are humans and there are no controversies about their identities. The main threat to these approaches in the case of human users, is a third party accidentally gaining knowledge of the confidential information. If we assume agents capable of split and merge, however, a large number of potential agents, obtained by splits and merges from the original agent can have access to this information. We can conclude that knowledge based authentication only is only a proof of membership in a group of descendants of the original agent, but not a way to uniquely identify an agent.

Another authentication approach is the use of biometric information, such as fingerprints, voiceprints, handwriting and face recognition. It is interesting to note that these approaches prove somatic identity. The equivalent case for software agents would be the proof that the agent is inhabiting the same software process, a feasible technical problem. Again, this approach fails for mobile or splitted agents.

C. Population

The problem of population refers to the number of individual persons which can share a single ”body”. Although the common wisdom says that a single body contains a single person, there are several facts which question this view. The controversial case of split personalities, for example, are not satisfying the memory criterion for personal identity - the individual personalities do not remember the actions of the other personality. On the other hand, we can argue that more subtle psychological ties always exists between the split personalities, so they are better considered as a single individual with selective amnesia than as multiple personalities.

Another argument is based on the relative independence of the two cerebral hemispheres. In fact, surgeons can cut the nerve bands between the two sides of the brain (*commissurotomy*, an operation frequently employed in the treatment of severe epileptics). Some researchers such as Dennett [11] are sugesting that there are two personalities in every human body. In a theoretical case, these personalities might be separated in two bodies, through a partial brain transplant.

For the case of software agents, the problem is more of a viewpoint. The action of merging agents has no equivalent in humans. Multiple, previously independent agents can be merged into a single *soma*, while continuing their independent actions. Depending on the internal implementation, we can choose for anything between the basically complete separation (no merging of knowledge or memories) to the

[4]We do not consider the software agents to have a ”personality” or a ”soul”.

complete merging of the agents. In practice, however, would be very difficult to ascertain the level of separation of different identities without a thorough understanding of the implementation details.

An additional technical problem is that modern software agents are usually implemented based on components [12], [13], and thus highly separable. In case of humans, arguments can be made that an individual brain hemisphere can sustain a personality, but the ability to separate stops here. In general it does not seem useful to consider an agent holding multiple identities just because it is formed of components which could be, theoretically, form the basis of an individual agent.

On the other hand, we consider useful at least tracing the identities of original agents in the merged agent. Our goal is not to capture the component structure of the agent, but to trace the fact that the ”experience” of the agent is coming through multiple life-lines.

D. Choosing answers

A researcher of software agents faces a different set of problems than a philosopher concerned with the human identity. The task is partially easier: we have the means to enforce the any consistent set of rules of identity in an agent society. There are a number of requirements against such a set of rules. We need to uniquely identify the agent responsible for any set of actions as well of the owner of given resources. The rules of managing the identity should fit as best as possible with our common sense understanding of identity. The rules can be externally enforced, but it is preferable that respecting the rules to be in the best interest of the individual agents and their human owners. This would be something like a ”constitution” or a ”basic set of laws” of the agent society [14]. The notion that the preservation of the identity is in the best interest of agents is not as obvious, as it seems. For example, Derek Parfit argues in [15], [16] that identity does not matter in the survival.

It seems obvious that the notion of the ownership and responsibility are tied to the identity. Resource ownership is a very broad term - from traditional computer resources such as bandwidth or processor time to resources traditionally associated with human persons - such as financial resources (for E-commerce agents). The responsibility for actions can also be expressed in terms of resources. Of course, for the foreseable future, the ultimate owner and responsible is the human owner of the agent, but this transfer of ownership and responsibility only happens in the last step and it does not remove the requirements of identity tracing.

One consequence of associating ownership and responsibility with identity is that we need to identify and enforce a unique set of rules for managing identity in every agent society. Disjoint agent systems might choose to apply different sets of rules, but the rules can not be mixed in a society of interacting agents. In fact, ”identity theft” can be a very frequent occurence in a large multi-agent system.

We also require the agents to conform (as much as possible) to the common sense understanding of the identity. The agent community has a tradition of introducing anthropomorphic notions in the design and deployment of agents. In many cases these do not correspond to the reality at the level of implementation (for example, in the case of mobile agents). There are however, obvious benefits for agents to adapt their abstractions to the *modus operandi* of their users. One of them is to ease the adoption process of new technologies. Another advantage is that the human laws regarding ownership and responsibility can be extended to the case of agents.

III. Practical problems of identity

The problem of identity is not one of only theoretical interest. It permeates the development process from design to implementation, from deployment to post-mortem traceback. The three important questions about the identity of an agent are: (a) How do I find the agent? (b) What are the rights of the agent? How can an agent prove that it has these rights? (c) Which agent is responsible for a given action? How can I prove it?

These problems are extensively covered by the agent, networking, distributed systems and security communities, a large number of solutions were proposed and they are in various stages of standardization. However, all these solutions assume that the identity of the agent is not in question. At most the problem is to prove the identity of an agent (authentication).

We argued in Section II of this paper, that the problem of the identity is not trivial if we allow for the splitting, merging and termination of agents. In this section we consider how these problems appear in implementation, and propose a provably consistent solution for handling them.

A. Implementation scenario. Split and merge.

First, we need to address an apparent contradiction: many successful multi-agent systems were designed without specifically addressing the problems of identity. Why would we need to care about it now? The answer is, that one can always find particular cases when the problems of identity can be simplified away. For example, we can use an agent system to perform a parallel computation on the grid. The master agent starts with a set of available resources. It spawns a number of new agents on grid locations, and delegates to them an amount of resources necessary to run. These new agents start with a new, unique identity which is unrelated to the master agent, and all of them run to their natural termination. The result of the computation is transferred back to the original location, and the master agent summarizes them. We can also make some simplifying assumptions regarding the responsibility of the agents: the agents are not responsible for any damage besides the full use of the allocated resources and the only responsible agent is the master agent.

Although the system used agent cloning and some of its components terminated earlier, it doesn't seem to raise any problem concerning agent identity. This apparent ease is due to a set of simplifying assumptions:

- The agent system has a simple, hierarchical organization. The organization of the agent system is known in advance. The life-cycle of the agent system is limited and known in advance. None of the agents have meaningful interaction outside the agent system.
- There was no merge operation. All the secondary agents terminated without being merged with each other or with the master agent.
- There is a single point where the rights and the responsibilities of the agents converge (at the master agent).
- The relationships between agents are simple master-slave relationships. There are no conflicts of interests, collaboration based on negotiation or multiple roles.

Let us reformulate our implementation scenario with an approach which does not take the simplifying assumptions. An agent is given the goal to give periodic, targetted weather predictions using resources on the Grid, such that it minimizes the cost of the execution. The agent is given an expense account, which he can use to buy computational capacity, sensor data and hire other agents to perform computations. The agent might meet other agents which have similar subgoals and can subcontract with them. As the load is not constant, the number of agents working on the task varies accordingly. The unnecessary agents are either terminated or merged. When needed, new agents are created through splitting or cloning. These splits happen based on local needs, not under the control of a centralized authority.

The lifetime of the agent system is indefinitely long. During this time, it is likely that a number of unexpected events would occur. Some agents might be terminated because of hardware or software failures. Their tasks need to be delegated to other agents, together with their resources and responsibilities. Some agents might be involved in actions with legal or financial consequences (for example, security breach on a computer because of a buffer overflow vulnerability in an agent). It is likely that in the near future this kind of activity would be punished by fines against the agents' resources. If since the offending event the agent terminated, split or merged, we need to identify the agent which is considered responsible according to an universally accepted rule.

B. Naming. Extended names.

We have seen that maintaining and establishing the identity of agents with the ability to split and merge is a non-trivial enterprise. The rules for assigning and reassigning identities should be unequivocal and simple. These rules should determine the names of the agents after the three major lifecycle events: termination, split and merge. We are proposing a naming scheme and associated ruleset which satisfies the aforementioned properties.

Definition 1: The agents maintain a data structure called **extended name** which is a triplet $(\mathscr{I}, S, \mathscr{C})$. The **identity set** $\mathscr{I}$ is an anordered set of identities $\{i_1, \dots i_n\}$, the **split number** S is a positive integer, while the **candidacy set** $\mathscr{C}$ is a set $\{c_1 \dots c_m\}$ of **candidacies**, where a candidacy is an ordered list $(i, n_0, n_1 \dots)$ with i an identity and n_i a positive integer.

The identity set determines the right and responsibilities of the agent. Any interaction of the agent is signed by one of its identities. Resources are allocated to a single identity. The agent's responsibilities and resources are the union of the responsibilities and resources of individual identities.

Rule 1: **Creation rule:** a newly created agent has the extended name $(\{i_{new}\}, 0, \emptyset)$.

Rule 2: **Split rule:** if agent $(\{i_1, \dots i_n\}, S, \{c_1 \dots c_m\})$ is split, the extended names of the two ancestor agents will be $(\{i_1 \dots i_n\}, S+1, \{c_1 \dots c_m\})$ and $(\{i_{new}\}, 0, \{(c_1 0), \dots (c_m, 0)\} \bigcup \{(i_1, 0) \dots (i_n, 0)\})$. We are using $(c_i, 0)$ as a shorthand for the list c_i extended with another 0.

Definition 2: We define the relation $\overset{c}{>}$ on candidacies for identity i as follows: $(i, n_0 \dots n_k) \overset{c}{<} (i, m_0 \dots n_l)$ if $i = j$ and any of the following conditions hold true:

(a) $n_0 < m_0$

(b) $k = 0$, $l > 0$ and $n_0 = m_0$

(c) $n_0 = m_0$ and $(i, n_1 \dots n_k) \overset{c}{>} (j, m_1 \dots n_l)$.

We define the relation $\overset{c}{\leq}$ by $x \overset{c}{\leq} y$ if and only if $y \overset{c}{>} x$ or $x = y$.

Property 1: The set of all possible candidacies for the identity i is a totally ordered set in respect to the relation $\overset{c}{\leq}$

For example $(A, 2) \overset{c}{\leq} (A, 3)$, $(A, 0, 2) \overset{c}{\leq} (A, 0, 3)$ and $A(0, 1, 2, 1) \overset{c}{>} A(0, 1, 2)$.

Rule 3: **Merge rule:** if the agents with the extended names $(\mathscr{I}_1, S_1, \mathscr{C}_1)$ and $(\mathscr{I}_2, S_2, \mathscr{C}_2)$ are merged, the extended name of the merged agent will be $(\mathscr{I}_1 \bigcup \mathscr{I}_2, max(S_1, S_2), \mathscr{C}_1) \bigcup \mathscr{C}_2$.

Rule 4: **Termination rule:** If an agent $(\{i_1 \dots i_k\}, S, \mathscr{C})$ terminates, for every identity $i \in \{i_1 \dots i_k\}$ a successor agent $(\mathscr{I}_s, S_s, \mathscr{C}_s = \{C_s(i) = (i, \dots)), \dots\})$ is selected such that its candidacy for the identity i, $C_s(i)$ is the maximum of the candidacies for i related to the order relation $\overset{c}{\leq}$. The new extended name of the candidate will be $(\mathscr{I}_s \bigcup i, max(S_s, S), \mathscr{C}_s \setminus C_s(i)\})$

For practical purposes, for the candidacy set of an agent we will retain only a single candidacy for every identity, the one which is the "greatest" according to the $\overset{c}{\leq}$ relation.

Rule 5: **Migration rule:** A migration operation of an agent is a split operation followed by the transfer of the agent holding the original identities to the remote location, while the agent holding the candidacies remains at the local site. If the successful migration is confirmed, the local agent is terminated.

We note that after a successful migration, the agent will have its split number incremented. In the case of failure, there are essentially two cases. In the first case the confirmation message was lost, but the remote agent was successfully created. We will have two agents, but no identity conflict: the holder of the identity is the agent at the destination site. The agent on the source site will actively search for the existence of the created agent and as soon as it got the confirmation that it exists, it will terminate. In the second failure scenario the remote agent is not created. When this is confirmed, the closest candidate will become the holder of the identity (according to the termination rule), and this will be the agent on the source site. Thus, Rule 5 prevents the confusion of the identities in the case of accidental cloning because of failed migration, but, of course, it can not prevent the identity theft by malicious cloning.

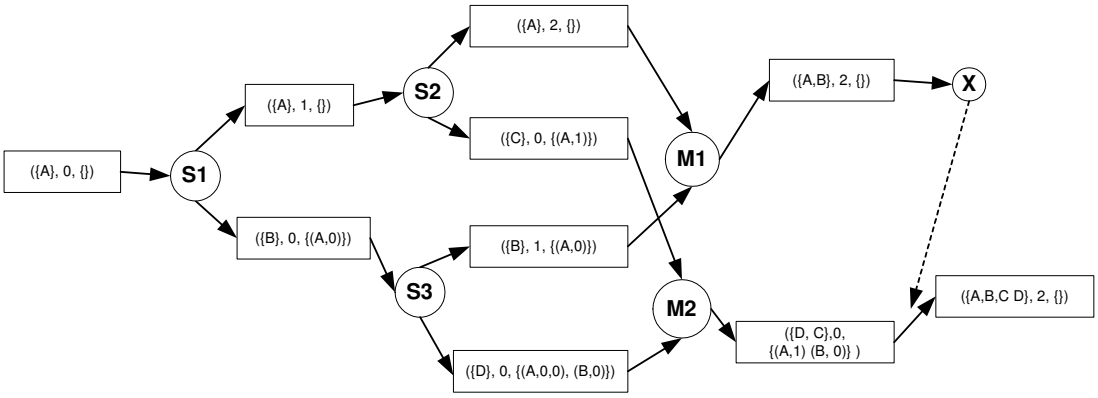

Fig. 3. An example of the evolution of the extended names after a series of splits, merges and an unexpected termination

C. Properties of the naming scheme

Property 2: For any two agents with the extended names $(\mathscr{I}_1, S_1, \mathscr{C}_1)$ and $(\mathscr{I}_2, S_2, \mathscr{C}_2)$, $\mathscr{I}_1 \bigcap \mathscr{I}_2 = \emptyset$

Property 3: For any two agents with the extended names $(\mathscr{I}_1, S_1, \mathscr{C}_1)$ and $(\mathscr{I}_2, S_2, \mathscr{C}_2)$, $\mathscr{C}_1 \bigcap \mathscr{C}_2 = \emptyset$

Property 4: Every identity is owned by a unique agent as long at least one descendant of the agent which initially owned the identity exists.

One interpretation of this property is the following: every action has a unique agent responsible for it, as long as one descentant of the agent commiting it exists.

Property 5: The continuations selected by rule 4 are the agents with the maximum possible common lifespan with the terminated agents.

Property 6: Let $(\mathscr{I}, S, \mathscr{C})$ a valid extended name at time t and $(\mathscr{I}_2, S, \mathscr{C}_2)$ an extended name at time $t + \Delta t$. If there was no agent termination in the interval $[t, t + \Delta t]$ then if $\mathscr{I} \bigcap \mathscr{I}_2 \neq \emptyset$ then $\mathscr{I} \subset \mathscr{I}_2$. The property does not hold if any of the agent $(\mathscr{I}, S, \mathscr{C})$ or any of its successor agents were terminated in the interval $[t, t + \Delta t]$.

We leave the proofs of these properties to the reader.

D. An example

Figure 3 shows an example of the evolution of the extended names of an agent after two levels of binary splits, two merge operations and a unexpected termination.

We start with the creation of the agent A, whose initial extended name is $(\{A\}, 0, \{\})$. This extended name is read as follows: it is the holder of the identity A, it had 0 splits, and it is not a candidate for any other identity. This agent is split into two parts in the operation marked S1 in the figure, whose extended names we obtain by applying the rule 2. From one of the agents, one is the continuation of the identity A, and it has the extended name $(\{A\}, 1, \{\})$, the only difference being the increment of the split number. The other agent will begin a new identity with a label generated to be unique. This identity, B, had no splits at this time. At the same time, this agent becomes a candidate for the identity of A. All this leads to the extended name $(\{B\}, 0, \{(A, 0)\})$.

Similar applications of the rule 2 lead us to the extended names of the four agents generated after the split operations S2 and S3. We should note that at this moment, there are four agents, all of them having a different identity A, B, C and D, and all the agents are candidates to the identity A of the original agent.

Merge operation M2 is merging the agents with the external names $(\{C\}, 0, \{(A, 1)\})$ and $(\{D\}, 0, \{(A, 0, 0), (B, 0)\})$. Applying the rule 3 we first infer that the merged agent will be a holder of both identities C and D. Its split number will be the maximum of the split numbers of the two entering agents, which is 0. The candidacy set of the merged agents will be the union of the two candidacies,

after the possible simplifications are applied. As $(A,1) \overset{C}{>} (A,0,0)$, we will retain only $(A,1)$ as the candidacy of the agent for identity A. As a result, the extended name of the merged agent will be $(\{D,C\},0,\{(A,1),(B,0)\})$.

The last event presented in figure 3 is the unplanned termination of the agent $(\{A,B\},2,\{\})$. At this moment, the identities A and B are not hold by any agent. The system is identifying agents which have the most justified candidacies to assume these identities. In this case, the only candidate is the agent $(\{D,C\},0,\{(A,1),(B,0)\})$. Applying the termination rules 4, this agent will be renamed as $(\{A,B,C,D\},2,\{\})$.

IV. Conclusions

In this article we have studied the problems of identity in software agents. Although personal identity is a relatively well researched problem in philosophy, the subject was not explored in agent research. We have shown that some of the theoretic thought experiments regarding personality fission are everyday reality in the world of software agents. Finally we have proposed an extended naming scheme through which the identity of agents can be traced through a sequence of split, merge and termination events.

The author is aware about the relatively controversial nature of the approach. Certainly, personal identity is just a mental tool through which the evolution of an agent society can be traced. However, it is our conviction that looking at the problems of identity offers a fresh and potentially fruitful perspective at the behavior of agent societies.

References

[1] O. Shehory, K. Sycara, P. Chalasani, and S. Jha, "Agent cloning: An approach to agent mobility and resource allocation," *IEEE Communications*, vol. 36, no. 7, pp. 58–67, July 1998.

[2] "Stanford encyclopedia of philosophy," URL `http:plato.stanford.edu`, 2003.

[3] S. Franklin and A. Graesser, "Is it an agent, or just a program?" in *Proceedings of the Third International Workshop on Agent Theories, Architectures and Languages*. Springer Verlag, 1996.

[4] J. Locke, "Of identity and diversity," in *Personal identity*, J. Perry, Ed. University of California Press, Dec 1975.

[5] S. Shoemaker, "Personal identity: A materialist's account," in *Personal identity*, S. Shoemaker and R. Swinburne, Eds., Dec 1984.

[6] M. Belzer, "Parfit and Lewis on Survival of Fission and What Matters in Survival: Why Parfit won the Debate," URL `http://personal.bgsu.edu/ mbelzer/parfitlewis.html`, 2003.

[7] N. R. Jennings, "Commitments and conventions: The foundation of coordination in multi-agent systems," *The Knowledge Engineering Review*, vol. 8, no. 3, pp. 223–250, 1993. [Online]. Available: citeseer.nj.nec.com/jennings93commitments.html

[8] F. Wan and M. P. Singh, "Commitments and causality for multiagent design," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. ACM Press, 2003, pp. 749–756.

[9] M. J. Kollingbaum, T. J. Norman, and C. Reed, "Implementing responsibility for states and events," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. ACM Press, 2003, pp. 1040–1041.

[10] W. Stallings, *Cryptography and Network Security - Principles and Practice*. Prentice Hall, 1999.

[11] D. C. Dannett, *Consciousness Explained*. Back Bay Books, 1992.

[12] C. Szyperski, *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley, 2002.

[13] L. Bölöni and D. C. Marinescu, "A component agent model - from theory to implementation," in *Second Intl. Symp. From Agent Theory to Agent Implementation in Proc. Cybernetics and Systems, Austrian Society of Cybernetic Studies*, April 2000, pp. 633–639.

[14] N. H. Minsky and V. Ungureanu, "Law-governed interaction: a coordination and control mechanism for heterogeneous distributed systems," *ACM Transactions on Software Engineering and Methodology*, vol. 9, no. 3, pp. 273–305, 2000. [Online]. Available: citeseer.nj.nec.com/minsky00lawgoverned.html

[15] D. Parfit, "Personal identity," *Philosophical Review*, vol. 80, pp. 3–27, 1971.

[16] ——, *Reasons and Persons*. Oxford University Press, 1984.