OpenSocial: From Social Networks to Social Ecosystem

Juliana Mitchell-Wong[1], Ryszard Kowalczyk[1], Albena Roshelova[1], Bruce Joy[2] and Henry Tsai[2]

[1]Centre for Information Technology Research, Swinburne University, Hawthorn, VIC, Australia
e-mail: (jmitchellwong, rkowalczyk, aroshelova)@ict.swin.edu.au
[2]Everyday Interactive Networks, Hawthorn, VIC, Australia, e-mail: (brucejoy, henrytsai)@ein.com.au

***Abstract*—Unlike the physical world where social ecosystems are formed from the integrated and managed relationships between individuals and organisations, the online digital world consists of many independent, isolated and incompatible social networks established by organisations that have overlapping and manually managed relationships. To bring the online digital world in-line with the physical world, integration of social networks, identification of overlapping relationships in social networks, and automation of relationship management in social networks are required. OpenSocial is a framework that enables social networks to interlink and self-organise into a social ecosystem guided by the policies of individuals and organisations.**

***Index Terms*—social framework, self-organised, self-managed, policies.**

I. Introduction

The relationships formed between individuals and organisations, and its management have created the social ecosystem that we live in today. The technologies available for establishing and maintaining this relationship have improved over time and the development of technologies in the online digital world is now upon us.

Currently the online digital world consists of many social networking communities (social networks with interacting tools and services) typically formed by organisations. These communities are independent, isolated and incompatible due to the lack of standard for forming social networks.

Many entities (individuals and organisations) have found that membership in a community limits the relationship they can form, that is to entities who are members of the community. It also limits the type of interaction to the tools and services provided by the community. The entities have found this to be too restrictive and not fully serving their needs.

This led organisations to establish their own communities, and individuals to become members of multiple communities. Over a period of time the number of communities increased, many providing similar tools and services, and had similar members. This soon became too time-consuming, complex and tedious for an individual to manage; and communities realised that forming alliances with other communities could improve the content and services it provides to its members.

Meanwhile, some individuals have resorted to actively participating in only a few communities, whilst there are others that use applications that enable different communities to be managed using the one application. GAIM[1] and Trillian[2] are two example applications for instant messaging communities. These applications however do not address any of the fundamental issues: the independent and isolated nature of communities, the ignorance to overlapping relationships in different communities, or the manual management of relationships.

Communities on the other hand have moved towards forming alliances with other communities to enable content search and retrieval between them by using common ontology [1]. The use of common ontology enables communities to interlink, but each of these communities assumes that their policies are agreeable by every community in the alliance. One issue that arises from this assumption is privacy. If a community shares personal and private information about its members to others in the alliance, it may lead to unsolicited interaction within the alliance.

This concern can be addressed by enabling entities to decide on the accessibility of its information by other entities, in other words an entity-centric identity management system such as Identity 2.0[3] and Liberty Alliance Federated Identity [2]. Both these standards aim to provide identity management in the online world that is similar to the drivers licence or passport in the physical world. This is where an independent identity provider issues an identifier about the entity, and other entities can accept it and authenticate the entity based on the identifier. Whobar[4] is a new Identity 2.0 technology that enables entities to register and login using identity protocols such as Windows CardSpace[5], i-names[6], and OpenID[7]. This enables the entity's identity to be distributed according to its preferences while enabling other entities to decide on the trustworthiness of the different authorities and their adequacy in providing authentication. However entities are still faced with many tedious tasks such as discovering entities with common interest, interacting, and managing relationships.

Automating these tasks can be achieved with policies defining the behaviour of agents based on the preferences of the entity [3-5]. This ease in discovering, interacting, and managing relationships in a social ecosystem makes it an ideal place to abuse the privacy and security of entities.

[1] http://gaim.sourceforge.net/
[2] http://ceruleanstudios.com/
[3] http://www.identity20.com/
[4] http://whobar.org/
[5] http://msdn.microsoft.com/winfx/reference/infocard/default.aspx
[6] http://www.inames.net/
[7] http://openid.net/

This abuse can be addressed by building a trust, reputation and relation system through the monitoring and management of interactions. This can be achieved by attaching sanctions or rewards to interaction behaviours.

In this paper we propose OpenSocial that aims to deliver an open framework that is scalable, interoperable, and inclusive of all existing communities to ensure rapid adoption. It supports a social ecosystem that interlinks individuals and organisations, and enables the automation and management of their social networks. It is set to change the Internet from a provider-centric (multiple consumers are related to a provider) to a consumer-centric (multiple providers are related to a consumer) paradigm leading to a change from information pull (consumer requesting) to push (provider predicting the consumer request).

The paper presents the OpenSocial framework with Section II describing the framework design and Section III detailing the lifecycle of automating an interaction request. Section IV demonstrates a case study for the framework and Section V describes an implementation of the framework. Finally the concluding remarks and future work are presented in Section VI.

II. Framework Design

The framework design is outlined in Fig. 1. It consists of three layers: connectivity, automation and management, and application.

A. Connectivity Layer

The foundational connectivity layer is built on a distributed or peer-to-peer (P2P) infrastructure with a peer representing an entity. There are many existing P2P routing protocols that can be implemented such as Gnutella, Kademlia [6] and Tapestry [7].

This infrastructure addresses issues in scalability and privacy. It enables new entities to be easily added to the ecosystem without great cost to any single entity in content storage, computational power, or network bandwidth, thus making it scalable. And it resolves privacy of information by allowing each entity to manage their profile.

The structure of this layer is modelled with the Agent Modelling Language (AML) [8, 9] and is shown in Fig. 2. The agents in this structure are entities of the ecosystem. Some of these entities may host communities with members consisting of other entities.

The functionality of this layer is to enable the linking of entities in the ecosystem.

B. Automation and Management Layer

The automation and management layer is built on top of the connectivity layer. This layer is represented by a number of intelligent software agents that automate the interaction requested by the entity and upholds the integrity of the social ecosystem according to the entity-defined policies. These policies ensure the agents perform to the requirements of the entity in an autonomous and consistently trustworthy manner; thus enabling the interlinking of entities across communities that would otherwise be too tedious and time-consuming to perform.

Automation agents include linkage to the ecosystem, search for entities, authentication of entities, authorisation of entities, comparison of identities, negotiation of policies, and the monitoring of interactions. Each of these agents has policies governing their automation.

Linking to the ecosystem involves selecting one entity that has a static IP address and is always available in the ecosystem, and linking to it. This in turn will provide links to other entities in the ecosystem. An AML model of the linking specification is shown in Fig. 3.

Searching for other entities involves finding the links representing the entities which include known entities, referred entities and randomly selected entities from the ecosystem.

Authenticating identifiers received from other entities involves sending the received identifier to the claimed identity provider to validate the authenticity of the identifier.

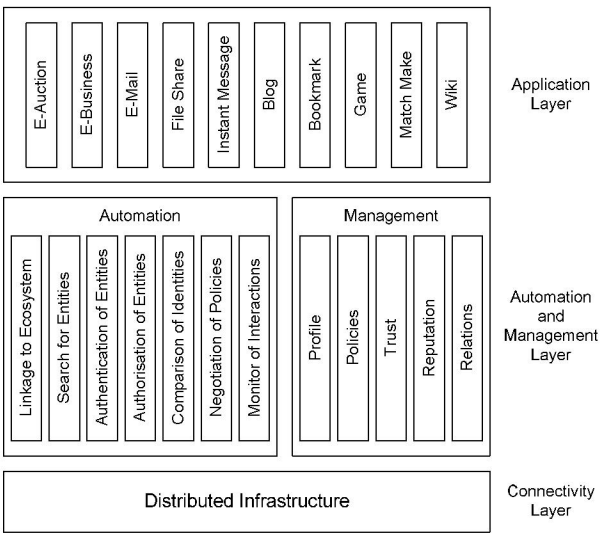

Fig.1 OpenSocial Framework design.

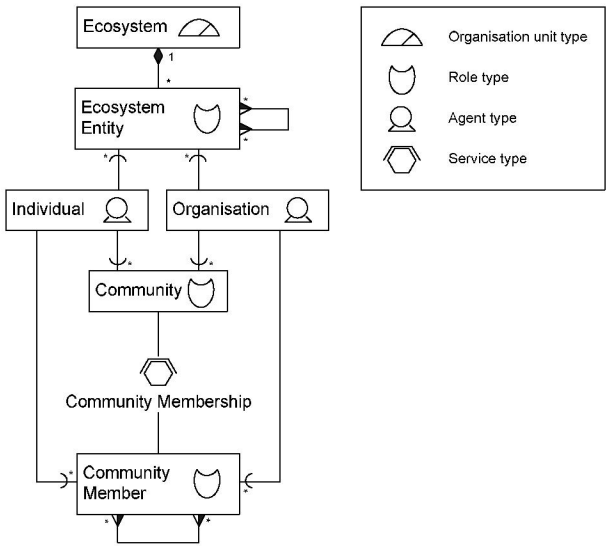

Fig.2 Ecosystem structure.

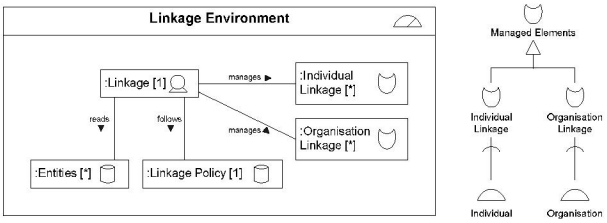

Fig.3 Linking specification.

Authorising other entities involves providing access to information or tools the other entities are entitled to.

Comparing the identities of other entities involves identifying similarities and differences in the identity of other entities to its requirements. There are numerous existing algorithms for performing this social comparison or matching [10-12].

Negotiating interaction activity policies with other entities involves reaching an agreement on the policy to use during the interaction.

Monitoring interactions with other entities involves analysing the communication messages sent and received.

Management agents include those for profile, policies, trust, reputation and relation. Each of these components invokes intelligent software agents to manage them according to policies which may be defined by an entity such as for its profile, its policies, its trust of another entity and its relation with another entity, or defined by a community such as the reputation of an entity. The location of the managed information may be central such as at the community, or distributed across entities in the community [13-15]. However it should not be located at the entity where it is the subject as it may lead to potential security breaches (data integrity).

The profile is used by many automation components such as logging in to communities, for the community information and the corresponding identifier used for login; and search for other entities, where the list of known and referred entities is stored. Information in the profile can also be requested by other entities with results varying depending on their authorisation. The profile manager processes the request, receipt and response messages according to its policy to ensure privacy and security of the profile.

There are two types of policies: authorisation mainly for automation, and obligation mainly for management. Irrespective of the policy type, the policies are managed to ensure that appropriate policies are applied to each situation, and conflicting policies are detected and resolved. The policy manager processes the request, receipt and response messages according to its policy to ensure completeness and validity of policies.

Trust defines the confidence the entity has of another entity (subject). It obtains the trust measure from the trust and reputation measure of other entities as well as from its interactions with the subject. The trust manager processes the request, receipt and response messages according to its policy to update the trust measure. PeerTrust [15] is one algorithm that measures trust from reputation.

Reputation also defines the confidence towards the entity (subject). However, the confidence is determined by a group of entities rather than by a single entity. It obtains the reputation measure from the interactions the group of entities has with the subject. The reputation agent processes the request, receipt and response messages according to its policy to update the reputation measure.

Relation differs from trust and reputation in that it quantifies the interaction with the entity (subject). It obtains the relation measure from its interactions with the subject. The relation manager processes the request, receipt and response messages according to its policy to update the relation measure.

It is important to distinguish between trust or reputation, and relation as the trust or reputation of an entity can exist without having a relation with it; and a good relation does not imply a high level of trust or reputation since the good relation may stem from being the only entity providing a specific service.

C. Application Layer

The application layer provides a set of interface specifications to support various types of applications. If it adheres to the interface specifications, it assures compliance and interoperability with other applications. This enables entities to continue using their familiar user interface for an application but with the ability to interact with others that may be using a different user interface.

III. LIFECYCLE MODEL

The lifecycle of automating an interaction request consists of two stages: discovery and interaction. Each stage is represented by an intelligent software agent that relies on automation and management agents. The AML class diagram of the lifecycle is shown in Fig. 4.

A. Discovery

The discovery stage determines the entities that are suitable to interact with. To achieve this, the discovery coordinator first establishes links with entities in the ecosystem, and then searches for specific entities from these links. Each of these entities is authenticated with their identifier, and the appropriate authority is assigned to it. The suitability of these entities is determined through the comparison of

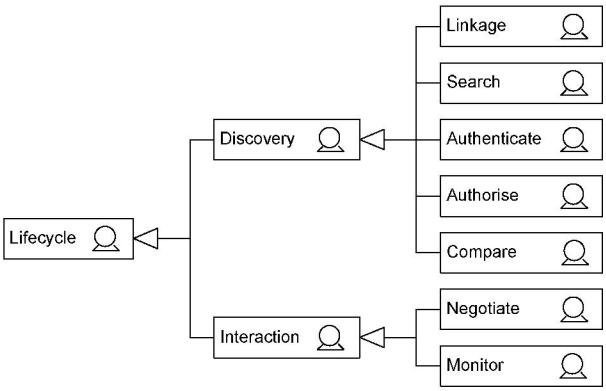

Fig.4 Lifecycle class diagram.

their identity with the requirements of interaction. These agent interactions are modelled using AML and shown in Fig. 5.

B. Interaction

The interaction stage determines the interaction activity process. To achieve this, the interaction co-ordinator negotiates with the other entity on the interaction activity policy. When an agreed policy is achieved, the entities begin to interact with each other. During the interaction, the communications between entities are monitored to ensure the integrity of the social ecosystem is upheld through updating their trust, reputation and relation.

IV. CASE STUDY

In this section a case study using the OpenSocial framework is demonstrated. There are three organisations: simpsons.com, garfield.com and garfieldfan.com, and nine individuals: Bart, Homer, Krusty, Jane, Peter, Tom, Jon, Garfield and Odie. Their relationships are shown in Fig. 6.

The relationships that the organisations have are:

- simpsons.com: Bart, Homer, Krusty
- garfield.com: garfieldfans.com, Jon, Garfield, Odie
- garfieldfans.com: garfield.com, Krusty Jane, Peter, Tom

The relationships that the individuals have are:

- Bart: simpsons.com, Homer, Krusty
- Homer: simpsons.com, Bart, Krusty
- Krusty: simpsons.com, garfield.com, garfieldfans.com
- Jane: garfieldfans.com, Krusty
- Peter: garfieldfans.com
- Tom: garfieldfans.com
- Jon: garfield.com, Garfield, Odie
- Garfield: garfield.com, Jon, Odie
- Odie: garfield.com, Jon, Garfield

This case study demonstrates the evolution of the ecosystem through Bart who is interested in finding trustworthy individuals in his local area (through advertising) that are interested in owning a kitten.

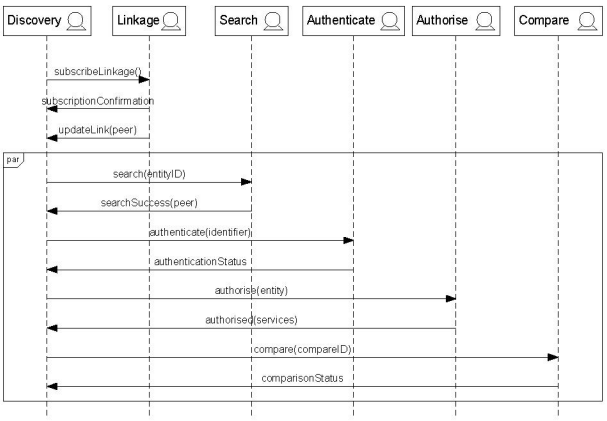

Fig.5 Agent interaction of the discovery co-ordinator.

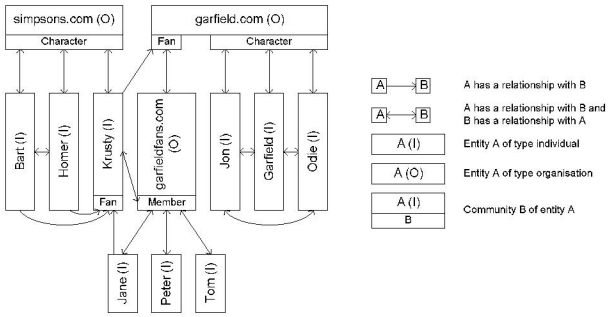

Fig. 6 Ecosystem of the case study.

A. Discovery

The first stage is for Bart to link to other entities in the ecosystem. In this case, he can form a link through simpsons.com, a permanent entity which has a static IP address and is always available online.

Through the simpsons.com community, Bart is linked to his father, Homer and to another simpsons.com character, Krusty. These are links of 2nd-degree.

3rd-degree links are new links created by either Homer or Krusty. Homer has links to simpson.com and Krusty but since Bart is already linked to both these entities, they are ignored. New links provided by Krusty includes garfield.com and garfieldfans.com

Bart is connected to every entity in this ecosystem after 4th-degree links. garfield.com provide new links to Jon, Garfield and Odie; while garfieldfans.com provide new links to Jane, Peter and Tom.

Next Bart searches for his 1st-degree relationships, Homer and Krusty from its links in the ecosystem. Once found they are authenticated, authorised and their identities are compared to determine their suitability for interaction. Homer and Krusty may also choose to refer other entities that may be suitable such as Krusty referring garfieldfans.com to Bart.

B. Interaction

The final stage is to interact with the discovered individuals. Bart's negotiation of the interaction activity policy with Homer led to an agreement that Homer would receive updates when any of the kittens have found owners. The negotiation with Krusty led to the agreement of a once-off advertisement, and with garfieldfans.com advertising to at least 25% of its members will ensure it is remunerated for its efforts.

These interactions are monitored such that if Homer receives an update that does not indicate any change to the kittens available, Bart is penalised in his trustworthiness and reputation, but his relation increases. Similarly if Krusty received the same advertising from Bart, Bart will be penalised in his trustworthiness and reputation but his relation increases. However, if garfieldfans.com manages to advertise to 50% of its members it would receive a financial benefit as well as an increase in trustworthiness, reputation and relation.

V. IMPLEMENTATION

The framework is implemented with Microsoft C# .NET. First, the 3 layers are integrated as illustrated in Fig. 7. The integration involves the Kademlia protocol for communication at the connectivity layer and provides an interface (Kademlia API) for the automation and management layer; the JADE platform[8] is used to develop agents that perform tasks for the automation and management layer and communicates using the JADE protocol; and the automation and management layer provides an interface (OpenSocial API) for the application layer that communicates with the application protocol.

The initial prototype has focused on integrating the 3 layers of the framework, creating placeholders for the agents, and enabling automation of the requested interaction through the 2 lifecycle stages. During this process, specifications for the profile and policies have been defined.

A snippet of the profile from the case study is shown in Fig. 8. The profile of an entity stores its identities, including information about the identity issuer and the location of the identity; its community memberships, including information about the community, the identity used to login to the community and its trust and relation with the community; its known entities, including information about the known entity's identity, list of referrers to the known entity, and the group it has categorised the known entity; and its trusted identity servers.

The policies are implemented using the open source implementation of the OASIS XACML 2.0 standard[9] in .NET[10] and a snippet from the case study is shown in Fig. 9. This snippet shows the policy for guiding the behaviour of the linkage agent. The rule states that only an entity that is a permanent entity is permitted to initiate a linkage. And once this permission is granted, the agent has the obligation to process the linkage through the permanent entity.

VI. CONCLUSION

This paper has presented OpenSocial, a framework that enables individuals and organisations to interlink across social networks and to self-organise into a social ecosystem guided by policies.

The initial prototype is implemented as a proof-of-concept and has focused on integrating the 3 layers of the framework and the agents in the automation and management layer.

Future prototypes aim to improve the automation and management agents in particular the negotiation and monitoring mechanisms. Negotiation can be improved by formalising the logic used in policies, whilst monitoring can be improved with algorithms that are specific to the social ecosystems environment.

[8] http://jade.tilab.com/
[9] http://oasis-open.org/committees/xacml/
[10] http://sourceforge.net/projects/mvpos/

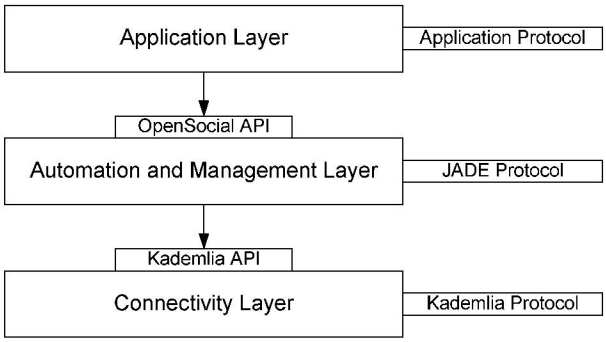

Fig.7 Implementation of the OpenSocial framework.

```
<Profile>
  <Identity>
    <Name>Simpsons</Name>
    <Server>
      http://www.simpsons.com/openid/server.bml
    </Server>
    <Identifier>
      http://bart.simpsons.com/
    </Identifier>
  </Identity>
  <Community>
    <Name>Simpsons Character</Name>
    <Server>
      http://www.simpsons.com/character/login/
    </Server>
    <Identity>Simpsons</Identity>
    <Trust>1</Trust>
    <Relation>1</Relation>
  </Community>
  .
  .
  .
  <KnownEntities>
    <EntityID>homer@simpsons.com</EntityID>
    <Identity>
      <Server>
        http://www.simpsons.com/openid/server.bml
      </Server>
      <Identifier>http://homer.simpsons.com/
      </Identifier>
    </Identity>
    <Referrer>
      <EntityID>
        http://www.simpsons.com/
      </EntityID>
      <Trust>0.8</Trust>
      <Relation>0.5</Relation>
    </Referrer>
    <Group>
      <Name>Family</Name>
      <Trust>0.7</Trust>
      <Relation>0.9</Relation>
    </Group>
  </KnownEntities>
  .
  .
  .
  <TrustedIdentityServer>
    http://www.simpsons.com/openid/server.bml
  </TrustedIdentityServer>
  .
  .
  .
  <PermanentEntity>
    <EntityID>http://www.simpsons.com/</EntityID>
  </PermanentEntity>
  .
  .
  .
</Profile>
```

Fig.8 Snippet of the case study profile.

```
<PolicySet xmlns:Profile="Profile.xsd"
           PolicySetId="OpenSocialPolicies"
           PolicyCombiningAlgId="deny-overrides">
  <Target/>
  <PolicySet PolicySetId="DiscoveryPolicies"
             PolicyCombiningAlgId=
               "deny-overrides">
    <Target/>
    <Policy PolicyId="Linkage Policy"
            RuleCombiningAlgId="deny-overrides">
      <Target/>
      <VariableDefinition
        VariableId="IsPermanentEntity">
        <Apply FunctionId="string-equal">
          <Apply FunctionId="string-one-and-only">
            <SubjectAttributeDesignator
              AttributeId="EntityID"
              DataType="#string" />
          </Apply>
          <Apply FunctionId="string-one-and-only">
            <AttributeSelector RequestContextPath=
"/Profile:Profile/Profile:PermanentEntity/text()"
              DataType="#string" />
          </Apply>
        </Apply>
      </VariableDefinition>
      <Rule RuleId="" Effect="Permit">
        <Description>
          Any entity in the network that is a
          permanent entity can initiate the
          linkage.
        </Description>
        <Target/>
        <Condition>
          <VariableReference
            VariableId="IsPermanentEntity"/>
        </Condition>
      </Rule>
      <Obligations>
        <Obligation ObligationId="Linkage"
                    FulfillOn="Permit">
          <AttributeAssignment
            AttributeId="Entity"
            DataType="#string">
            <SubjectAttributeDesignator
              AttributeId="EntityID"
              DataType="#string" />
          </AttributeAssignment>
        </Obligation>
      </Obligations>
    </Policy>
    .
    .
    .
  </PolicySet>
  .
  .
  .
</PolicySet>
```

Fig. 9 Snippet of the case study policies.

VII. Acknowledgements

This work has been supported by the Agent-Enabled Social Networks project in collaboration with Everyday Interactive Networks under the Australian Research Council's Linkage funding scheme (ARC Linkage Project LP0562500).

VIII. References

[1] J. G. Breslin, A. Harth, U. Bojars, and S. Decker, "Towards Semantically-Interlinked Online Communities," in 2nd European Semantic Web Conference (ESWC), Heraklion, Crete, Greece, 2005.

[2] Liberty Alliance Project, "Personal Identity," http://www.projectliberty.org/resources/whitepapers/Personal_Identity.pdf.

[3] G. Boella and L. van der Torre, "Local vs Global Policies and Centralized vs Decentralized Control in Virtual Communities of Agents," in IEEE/WIC/ACM International conference on Web Intelligence (WI), 2004.

[4] R. Gavriloaie, W. Nejdl, D. Olmedilla, K. E. Seamons, and M. Winslett, "No Registration Needed: How to Use Declarative Policies and Negotiation to Access Sensitive Resources on the Semantic Web," in European Semantic Web Symposium, Heraklion, Greece, 2004.

[5] S. L. Keoh, E. Lupu, and M. Sloman, "PEACE: A Policy-Based Establishment of Ad-hoc Communities," in Annual Computer Security Applications Conference (ACSAC), 2004.

[6] P. Maymounkov and D. Mazieres, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric," in 1st International Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, USA, 2002.

[7] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz, "Tapestry: A Resilient Global-Scale Overlay for Service Deployment," *IEEE Journal on Selected Areas in Communications*, vol. 22, pp. 41-53, 2004.

[8] I. Trencansky, R. Cervenka, and D. Greenwood, "Applying a UML-Based Agent Modeling Language to the Autonomic Computing Domain," in Companion to the 21st ACM SIGPLAN Conference on Object Oriented Programming Languages, Systems and Applications, Portland, Oregon, USA, 2006.

[9] R. Cervenka, I. Trencansky, and M. Calisti, "Modeling Social Aspects of Multiagent Systems: the AML Approach," in 6th International Workshop on Agent Oriented Software Engineering (AOSE), Utrecht, The Netherlands, 2005.

[10] L. Terveen and D. W. McDonald, "Social Matching: A Framework and Research Agenda," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 12, pp. 401-434, 2005.

[11] L. N. Foner, "Yenta: A Multi-Agent, Referral Based Matchmaking System," in 1st International Conference on Autonomous Agents, Marina del Rey, California, USA, 1997.

[12] H. Kautz, B. Selman, and M. Shah, "Referral Web: Combining Social Networks and Collaborative Filtering," in *Communications of the ACM*, vol. 40, 1997, pp. 63-65.

[13] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The EigenTrust Algorithm for Reputation Management in P2P Networks,"in 12th International Conference on World Wide Web, Budapest, Hungary, 2003.

[14] P. Dewan and P. Dasgupta, "PRIDE: Peer-to-Peer Reputation Infrastructure for Decentralized Environments," in *13th International Conference on World Wide Web*. New York, USA: 2004.

[15] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, pp. 843-857, 2004.