Combining reactive & deliberative agents for complete ecosystems in infospheres

Fabien L. Gandon
Mobile Commerce Laboratory, School of Computer Science, Carnegie Mellon University
Fabien.Gandon@cs.cmu.edu

Abstract

The diversity of resources and information in real infospheres calls for artificial ecosystems with a diversity of interacting agents ranging from reactive to deliberative paradigms and maintaining the information ecology. After discussing the notion of infosphere and some interests of the XML family for such a world, this paper provides examples showing the interest of such hybrid systems.

1. Notion of infosphere

The advent of information networks makes the cyberspace of William Gibson look like a clever anticipation more than a fiction. However, the effective result is a world '*wild*' web, where information resources and services are scattered, ever changing and growing; this makes it more and more difficult for humans to locate and access a relevant resource. Because these wild information landscapes are unorganized and heterogeneous in their form, content and quality, it is extremely difficult to automate tasks and provide intelligent assisting tools. In fact, from many perspectives, these landscapes can be compared to our own world: they are vast, distributed, heterogeneous landscapes; they provide a rich fertile soil of information resources; actors of these spaces can be situated inside; actors can perceive and act with their local resources; actors can interact through their environment.

From this similitude stems the metaphor of the infosphere: the equivalent in information worlds of our biosphere and its ecology. The biosphere is the sphere of *action* of life on Earth that encompasses the living *beings* together with their *environment*. It is a closed *ecosystem*, *self-regulating* through complex cycles involving *multiple interactions* within a *huge variety of living beings*, and between them and a *huge variety of environments*. Thus the main idea is to setup ecosystems in information spheres or *infospheres*: "the infosphere is the environment constituted by the totality of information entities - including all agents - processes, their proprieties and mutual relations." [1]

Distributed artificial intelligence is developing information agents in order to populate infospheres. Agents are defined as clearly identifiable individual artificial entities with well-defined boundaries and interfaces. They are situated in an environment they perceive through sensors, and act upon and react to through effectors. They have social abilities to interact with other agents or humans but meanwhile they keep their self-control over their behavior. Unlike humans, agents have infinite patience and perseverance, and they can exploit and manage huge amounts of information extremely rapidly. The importance and interest of a convergence between agents and web communities is now acknowledged [2]. Thus, in parallel to information agent development and to allow the automation of tasks, the W3C issued recommendations to bring structure (XML) and semantics (RDF/S and OWL) to the Web [3]. Together, both domains contribute to the development of complete infospheres.

Many people are already trying to build the economy, the ethic, the trust, *etc.* of these infospheres, but it is *still missing a stable ecology*. In fact, instead of ecology, current studies in the agent field rather look like autecology (*i.e.*, the study of one individual organism or one single species) while we really should move toward synecology (*i.e.*, the study of the ecological interrelationships among communities and species of organisms) and real ecology (*i.e.*, the study of the relationships of communities and species of organisms to their physical environment and to one another). A symptom of this lack is that there remains a dichotomy between intelligent information agents (as presented in [4]) and fine-grained agents [5] forming swarm intelligence [6] in information spaces as in Anthill [7] and its Gnutant application.

For most problems, neither a purely deliberative nor a purely reactive architecture is appropriate, and usual approaches in conciliating reactive and deliberative behaviors are at the agent architectural level [8], leading to hybrid agents with an architecture handling both reactive and deliberative behaviors, and usually based on either hierarchical or parallel layered architecture. While it is true that a human body is composed of cells that can be seen as smaller organisms and that, therefore, a holonic perspective of agents may be interesting, it is also true that humans are not composed of insects while they *do* benefit from the numerous ecological roles insects play and vice-versa. The organizational metaphor must be extended include hybrid complex systems composed of heterogeneous agents and organizations.

The complexity of the information spaces will call for complex regulating systems and new approaches such as autonomic computing [20] or ecology of infospheres. We should pursue the development of climax communities of information agents *i.e.* communities that can reach a stable

stage through a process of succession, whereby relatively simple communities provide a basis for more complex one: the idea is to develop in information worlds the counter-part of, for instance, food chains and food webs (*i.e.*, overlapping chains) to build information chains and webs providing at the end, great added value, compared to the initial fertile but wild information ground.

The diversity of infospheres shows there is need for a large spectrum of agent types (from purely reactive to complex deliberative agents) addressing the large spectrum of information tasks and services, and raising the question of the cohabitation and interactions between them, and between them and their environment. Our own ecology requires a full spectrum of beings and organizations of life; it is in complex equilibrium based on direct relationships (e.g. prey-predator) or indirect chains (e.g. insects degrade organic detritus into fertile mater, plants use this fertile soil to grow vegetal mater through photosynthesis, herbivores eat these plants to grow animal organic mater, *etc.*) Likewise, many forms of interactions can be envisaged between many different species of information agents. But currently, interactions are usually taking place within a single family of agent: stigmergy [9] (reactive agents communicating by modifying their local environment), communication at knowledge level [10] (deliberative agent communicating with languages, ontologies and protocols) and holonic approaches (where holons form communities which are reified as agents able to form new communities). A *complex information ecosystem includes chains and webs transcending families of agents* to build stable cycles and maintain the pyramid of species where each level brings some added value (more structure, analysis results, *etc.*) and helps extract, refine, exploit and manage the rich ore present in the information resources.

In the following, we shall focus on a special technological stance: XML information landscapes. We will then present two perspectives of interactions:

- The holonic customization of a behavior, where atomic tasks participating to the plan of an agent are externally scripted and exchanged as simple reactive agents.
- The farming of populations of reactive agents by intelligent agents to propagate tasks over the network.

2. Information agents and XML sphere

2.1. XML: description of information landscapes

XML is a description language recommended by the W3C [3] to define markup languages that describe structured document and data in text format, so that they can be used over internet-based networks and in particular over the Web. XML documents contain their own structure within themselves and parsers can access it and retrieve data from it. It is platform-independent and supports internationalization and localization. It makes it possible to deliver information to distributed software agents in a form that allows further processing and therefore distribute tasks. The set of elements, attributes, entities and notations that can be used within an XML document can optionally be formally defined in an XML Schema allowing validation in exchanges. XML is license-free and well-supported with a large set of tools and API. XML is also more and more used in commercial applications allowing tool-independence in data-flows and durable storage.

"The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation." [2] To do so, Resource Description Framework (RDF) [3] uses a triple model and an XML syntax to represent properties of Web resources and their relationships in what we call RDF annotations. It makes no assumption about a particular application domain, and annotations are either internal or external to the resources, thus existing documents may be kept intact and annotated externally. RDF is officially recognized as an effort of the W3C to integrate applications and agents into one Semantic Web [2]. Just like people need to have agreement on the meanings of the words they use, computers need mechanisms for agreeing on the meanings of metadata in order to communicate effectively. Formal descriptions of terms are called ontologies and are formalized and shared thanks to RDF Schema (RDFS) which is related to object models but with the properties being defined separately. The framework is designed to be extended in layers and the next one will be OWL [3].

2.2. XSLT: acting on information landscapes

Beyond XML, a family of extensions and modules is growing among which XSLT (Extensible stylesheet language transformation) is of special interest to us: it enables XML tree transformation into other XML trees or text. It is possible, for example, to generate a table of contents, adapt sorting of lists, *etc.* Thus a document can be viewed differently and transformed into other documents so as to adapt to the needs and the profile of the agents and the users while being stored and transferred in a unique format.

XSLT is a rule-based language where formatting rules, called templates, transform a source tree into a result tree. The transformation is achieved by matching template patterns against the source tree and instantiating template content to create the result tree. More than one template rule may have a pattern that matches a given element, but only the template with the most precise pattern will be applied. Operators enable to access the values of nodes, and branching instructions are available. Templates are applied recursively on the XML document, by finding the templates matching the children of the current node and applying them. There are facilities for sorting and counting elements, importing stylesheets, defining variables and parameters, calling templates by name and passing parameters.

The patterns of the templates and the tests in branching instructions use XPath, a language that enables to describe a path in an XML structure, express a selection pattern and retrieve element values. A path is composed of steps such as '/book/introduction' which denotes the 'introduction' child tags of the 'book' elements at the root of the document. Paths can include conditions and the result is the set of nodes satisfying this selection pattern. The path and conditions are expressed on axes that are navigation directions from a node to another, e.g.: ancestor. Functions are used to build selection paths and manipulate values.

In [11], a formal model of a subset of XSLT is analyzed and authors show that from a language theoretic point of view, XSLT expressiveness correspond to tree-walking tree transducers with registers and that its expressiveness is better than a number of XML query languages. Moreover, XSLT provides two ways of extension: one for extending the set of instruction elements used in templates and one for extending the set of functions used in XPath expressions. For these reasons, and because XSLT is part of the XML standards family, we use it to create and deploy simple script agents, as explained in the following sections.

2.3. Two agent perspectives for XSLT & XML

Many languages exist for mobile agent and scripts, ranging from Java that simply provides dynamic class loading, to one of the oldest and most well-known complete platform: Telescript from General Magic Inc. However it is out of the scope of this article to compare the different contributions. Suffice to say, that with XML becoming a universal exchange format for data, XSLT is becoming a universal exchange format for data manipulation: XML has been used to provide a declarative language for agent communication; XSLT can be used to provide a platform-independent procedural language in agent communication

Reactive and deliberative information agents have in common to be clearly identifiable individual artificial entities, situated in an information space, such as a network, that they can sense, react to and act upon. Reactive agent interactions usually rely on simple signal modifying the environment. Deliberative agents may communicate at knowledge level. Both have self-control over their behavior. The behavior of a reactive agent is usually simple and based on reflex mechanisms *i.e.* automatic reaction to a stimulus. While the deliberative agents may involve more complex behaviors (knowledge-based systems, BDI, machine learning techniques, *etc.*) they are also ultimately composed of a (planned) sequence of simple tasks. Based on these distinctions, we will envisage here two perspectives on the use of XSLT templates in the interactions of a multi-agent information system exploiting XML; in both cases XSLT is used to propagate simple XML manipulation behaviors.

The first option is to use XSLT scripts to dynamically customize generic information agent roles at run-time. Some of the atomic actions forming the plans and behaviors of deliberative information agents can be externally described by XSLT templates and exchanged between agents making them easy to customize and maintain in order to adapt to the users and the evolutions of the information resources. The behaviors of the intelligent agent can thus be designed at a fairly generic level of actions, relying on XSLT templates for final tuning. This perspective can be seen as a holonic approach where the overall behavior of a deliberative information agent relies on the intelligent composition of some simpler agents.

Technically speaking, the second option is equivalent. However, from a conceptual point of view, it is closer to the vision of pyramid of species where agents exploit the results and even farm agents of other layers to maintain the ecology of the infosphere. As illustrated in Figure 1, XSLT proposes some interesting constructors to describe and propagate simple reactive agents:

- *Sensors* are provided by the patterns of a template or the test instructions both using the XPath expressions.
- *Effectors* are the value-manipulating instructions that allow the agent to create the result tree.
- *Reactions* are encoded through recursive calls and branching instructions.

As an example, pheromones used in stygmergy can be implemented using special XML tags added or modified in the output of a template and sensed by the patterns of the same template or another one.

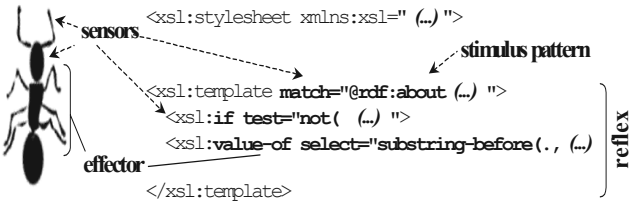

Figure 1. Anatomy of XSLT reactive information agent

The execution principle for these agents is that any XML piece they encounter is replaced by the output of their template. Therefore, the Rule #1 of such an agent is to respect its environment by leaving untouched the XML pieces it is not interested in modifying.

Thus any reactive agent has at least the following template that, by default, applies to any node or attribute and fully copies it to the output:

```
<xsl:template match="@*|node()">
  <xsl:copy>
    <xsl:apply-templates select="@*|node()"/>
  </xsl:copy>
</xsl:template>
```

Figure 2. Rule #1 for information preservation

If an agent only has this template, then it will run through XML documents leaving them unchanged. Since an XSLT engine always chooses the most precise template, the default template of Rule #1 ensures that if an agent has nothing better to do, it will at least respect its information environment. This rule can be enforced by intelligent information agents, for instance to prevent any intentional or accidentally harmful agents to propagate deletion. Additional more precise rules are then added to customize the reactions of this population of swarming agents. These templates are generated by intelligent information agents to propagate actions they are interested in (maintenance task, recurrent updates, *etc*.)

Generic protocols are used in both perspectives. For instance and as shown the following examples, the FIPA-Request protocol can be used to propagate a swarm population, create an information agent providing a given template as a parameter, trigger template updates, *etc.*

In both cases, we can associate metadata to templates, using RDF to declare the type of agent, identify the sender for security checks (the template can be encoded using a private key corresponding to the public one of the sender), provide parameters to be used when calling the stylesheet, manage a "time to live", *etc.* Several projects looked at XML as a structured world for intelligent agents to manage and for swarming agents to crawl; we go further:

- We use RDF to semantically annotate documents and allow agents to reason at the knowledge level.
- We use XSLT to describe processes exploiting the semantic annotations and to propagate tasks.

3. Experimentations

The following examples are based on two large projects. In the European project CoMMA [12] we implemented and tested a corporate memory management framework based on agent technology. We studied the materialization of a corporate memory as a corporate semantic web that comprises an ontology encoded in RDFS, descriptions of the organizational reality encoded as RDF annotations about groups (corporate model) and persons (user profiles), and RDF annotations about the documentary resources. The result of this approach is a heterogeneous and distributed information landscape, semantically annotated using the conceptual primitives provided by the ontology. The multi-agent system manages annotations about documents referenced by their URI. It focused on three functionalities: (a) improve precision and recall retrieving documents using semantic annotations; (b) proactively push information using organization and user models; (c) archive newly submitted annotations. The system was initially divided into four dedicated sub-societies of intelligent agents:

- The *user-dedicated sub-society* comprises three roles: Graphic User Interface Controller; User Profile Manager; User Profile Archivist
- The *connection sub-society* with two roles defined in FIPA: Agent Management System; Directory Facilitator
- The *society dedicated to ontology and model* comprises two roles: Ontology Archivist; Corporate Model Archivist
- The *annotation-dedicated society* comprises two roles: Annotation Archivist; Annotation Mediator

These roles mostly correspond to intelligent information agents and are implemented using knowledge-based systems (Annotation Archivist and Annotation Mediator use CORESE API [19]) or machine learning techniques (User Profile Manager).

In the *my*Campus project [13], supported by the DARPA under the DAML initiative, we are interested in an open architecture to support mobile accesses to context-aware services. Its current specifications revolve around a growing collection of Task-Specific Agents that users can contract and a collection of e-Wallet Agents (one per user) that provide a unified semantic and secured interface to knowledge about their owner. A User Interaction Manager is in charge of the interfaces and interactions with the users and a Platform Manager provides the administration facilities and maintains white and yellow pages. Agents use Semantic Web frameworks and Web Services to perform their tasks. The first experiment had two task-specific agents: a Restaurant Concierge Agent using food preferences and location tracking to make recommendations; a Message Filtering Agent using interests and activity status of its user to plan delivery.

3.1. Customize behaviors of information agents

3.1.1. Web wrappers. No organization is an island; it is included in a culture, a country, a society, a market, *etc.* and a lot of interesting information is available on the open-Web, relevant to the organization's environment, core activities, domain, *etc.* Being relevant to the organization, these resources can be annotated to integrate the corporate memory. However, the task of annotating a resource can quickly become tedious and repetitive. Some Web sites having a rather static structure, they provide structural clues (header, table, separator, *etc.*) that can be exploited to automate some extraction rules and enable the user to automatically generate annotations from the content of the resource. Thus we introduced a society of wrappers [14] to automate the extraction of relevant pieces of information and their integration to the organizational memory. The use scenario is as follows:

(1) Through their personal interface agent, the users indicate a sample Web page among the set of pages to be annotated and which have a similar structure. The agent retrieves the HTML source code, cleans it and converts it into XHTML to obtain a well-formed XML document. The user annotates the page through a graphic interface making extensive use of drag-and-drop actions from the tree structure of the page and the ontology O'CoMMA to

structure of the annotation. The agent internally derives XSLT data extraction rules. Some built-in templates provide high level extraction functions such as: recursive extraction of a list of data delimited by a given separator (e.g., for the list of authors), or replacing some data extracted by a corresponding concept in the ontology (e.g., for keywords). These built-in templates are transparent to the users, and are embedded in the overall extraction template, to provide an autonomous script for generating the RDF annotations of a targeted Web site.

(2) Once the template is validated by the user, the interface agent contacts a Wrapper Manager and requires the creation of a new Annotation Wrapper Archivist to handle this new source of annotations. The interface agent then sends the template and the location of the Web pages to the newly created wrapper.

(3) The Annotation Wrapper Archivist applies the XSLT Template using a standard XSLT engine to create the annotations for the whole site and archive them.

(5) Finally, the Annotation Wrapper Archivist registers with an Annotation Mediator like any Annotation Archivist, so that it is ready to participate to query solving. It also maintains its base, monitoring changes in the source.

This scenario allows users to develop and launch a population of wrappers monitoring assigned sources. The initial behavior of the wrapper includes functions such as web site monitoring, ontology-based search in annotations for query solving, annotation generation, *etc.* the latter is a generic task that is described at run-time by the XSLT template sent by the interface agent.

3.1.2. Semantic gateway. No organization is an island and, for its activity, it has to cooperate with other organizations; this leads to the creation of temporary extranets supporting a virtual organization. An interesting problem is then to allow the connection of the semantic intrawebs of the different organizations, setting up focused gateways for the time of the cooperation. One of the problems raised by such a gateway is the mapping between the ontologies of the different organizations. We designed a simple semi-automatic ontology mapping process we called tf*icf. It is an adapted version of tf*idf, for ontologies in RDFS having a good documentation in terms of labels and comments for every classes and properties. The system uses "term frequency inverse class frequency" to propose some mapping between the classes of two schemata and the user is able to visualize and validate these rules through a graphical interface in an incremental fashion. The idea we are pursuing is to derive from this mapping a set of XSLT templates that are able to translate a query or an annotation from a semantic intraweb to another one. As in the previous case, a Gateway Mediator originally has a generic behavior, and users can specialize and update it at will through the upload of XSLT templates.

3.1.3. Dynamic interfaces. Information adaptation is not the only source of tasks where this approach can be used. Interfaces can use the same approach to present information to a user applying customizing stylesheets [12]. Among the one we tested two are of interest here:

- Personalized simple query generation: a template uses users' profile (RDF annotation about users) and the ontology in RDFS to generate a portal proposing personalized and filtered views to browse the ontology and suggesting elementary queries.
- Result presentation: a template currently uses the results of a query in RDF and the ontology in RDFS to display and document the results in natural language.

Figure 3. Adapting interfaces to users

In both cases the use of XSLT templates allows the system to propagate updates (improvements, maintenance, etc.) and can also allow the users to customize them if needed. These abilities are even more vital in the *my*Campus project, where the family of Task-Specific Agents is open to enable new services to be added. These new services require new interfaces.

3.2. Swarm propagation of ecological tasks

3.2.1. Maintenance. The inevitable problems raised by the life-cycle of knowledge are an endless source of examples where the propagation of simple reactive agents can be useful. The problems require tasks such as: correction of annotation content in the archives, maintenance of the coherence of annotations after a change in the ontology, erasing old annotations, *etc.*

As an example, let us consider the case where the update is due to the change of the URI of a resource. Figure 4 presents three frames: an example of annotation to be corrected; the XSLT core for correction; the annotation after correction. This fairly simple case illustrates the use of a specialized type of reactive agents, tailored by an information agent to propagate a task over the whole memory. A population of such XSLT agents can be generated automatically by an intelligent agent or through interface manipulation and generic libraries of templates.

```
(…)
 <CoMMA:WebPage rdf:about="http://www.inria.fr/acacia/ ">
  <CoMMA:Title>Web page of ACACIA</CoMMA:Title>
  <CoMMA:CreatedBy>
   <CoMMA:Person rdf:about="http://www.inria.fr/people#rdieng/" />
  </CoMMA:CreatedBy>
 </CoMMA:WebPage>
(…)
```

```
(…)
<xsl:template match="@rdf:about[contains(.,'www.inria.fr')]">
<xsl:attribute name="rdf:about">
<xsl:value-of select="substring-before(.,'www.inria.fr')"/>
www-sop.inria.fr
<xsl:value-of select="substring-after(.,'www.inria.fr')"/>
</xsl:attribute>
</xsl:template>
(…)
```

```
(…)
<CoMMA:WebPage rdf:about="http:// www-sop.inria.fr/acacia/ ">
  <CoMMA:Title>Web page of ACACIA</CoMMA:Title>
  <CoMMA:CreatedBy>
   <CoMMA:Person rdf:about="http:// www-sop.inria.fr/people#rdieng/" />
  </CoMMA:CreatedBy>
 </CoMMA:WebPage>
(…)
```

Figure 4. XSLT agent propagating the update of a URI

3.2.2. Fermentation. Swarm intelligence is the property of a society where individual unsophisticated behaviors interact locally with their environment, causing a coherent functional global pattern to emerge. Using several populations of such agents and stygmergy mechanisms one can obtain complex results: a population can detect and complete an aspect, and another one detects this new addition and combines it with other aspects to produce another result and so on. For instance to support different stages of an information workflow one species can be generated for each step and maintain a link in the chains and webs of information workflows. *In knowledge rich spaces, swarm agents can maintain information fermentation* (shallow parsing, shallow pre-processing) to derive basic knowledge patterns.

Thus, this last example illustrates how reactive agents can be used to foster the emergence of simple consultation paths, using a see-also relation to build tracks and clusters of related resources. Given an annotation about a document, a new XSLT script can be generated to propagate pointers toward this resource in related annotations. This script tests if an annotation concerns a resource that has the same keywords, authors, *etc.* that the one it was generated from. This reactive agent is then propagated to enrich annotations with see-also suggestions. When the user consults one of these modified annotations, the system can display the associated suggestions and thus the user can start to follow a track of interest built by the deployment of several species of these swarm agents; in this way the system can suggest browsing paths or queries that may not have come to mind to the user or that are so usual that the provision of such short-cut is natural. The same procedure can be used to link like-minded people, build acquaintance networks, Web rings, *etc.* and also to suggest propagation paths in systems where query solving is based on query propagation or distributed indexing.

For testing, we used annotations extracted from the PubMed [15] using the wrapper previously described. PubMed is a service of the National Library of Medicine that provides access to over 12 million MEDLINE citations and additional life science journals. The test base contains 9981 extracted annotations and the behavior tested was the cross-pollenizing (call it bee2bee :-)) of annotations: an XSLT scripts starts from an annotation about a report and extracts its list of authors, then for each other annotation it visits, it leaves a pheromone if the visited annotation shares authors with the initial one:

```
<c:ResearchReport rdf:about="URL in visited annotation">
  (…)
 <c:sameAuthorAs>
  <c:SameAuthorDoc c:nbSharedAuthors="nb shared authors"
                   rdf:about="url initial document"/>
 </c:sameAuthorAs>
</c:ResearchReport>
```

Figure 5. Leaving a 'see-also' pheromone

This is a kind of "see also" link, that points back to the initial annotation and gives the number of shared authors which is used both to rank the pointers and to limit the list of pointers to the most relevant ones. Over the 9981 annotations, 7724 'sameAuthorAs' links were generated by pollenizing, linking 2728 reports together *i.e.* 27% of the base. Figure 6 shows the number of reports *versus* the number of authors they share using logarithmic axes; as it is usually the case in biology, the list of authors for a publication can be very long and, as shown on the figure, two of the reports actually shared 191 authors.

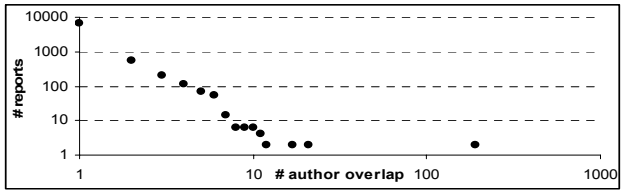

Figure 6. Number of reports *vs.* overlap in authors

In this last example we see a complete cycle involving several species of agents: the wrapper is a software agent that was customized to extract the annotations; the 'pollenizing agent' is a reactive agent enriching the annotations; the Archivist Agent maintains the bases of annotations and answers to queries as much as it can with its local resources; the Annotation Mediator manages the querying process distributed among the archivists and achieves the result integration. Thus when a user submits a query, its solving effectively leverages the work achieved by the whole chain of agents of the infosphere.

4. Discussions and perspectives

Several needs were experienced in using XSLT 1.0 and some of them will be addressed in XSLT 2.0 (in particular conversion of result tree fragments to node-sets, multiple output documents, built-in support for grouping, user-defined functions). Also, XPath 2.0 will provide an expression language for processing sequences and it introduces support for the XML Schema primitive types. However, this article was not about the expressiveness of XSLT, and did not claim XSLT was enough for every type of reactive agent.

Rather, this paper showed that there is need for every type of agent and it defended the idea of going beyond the organizational approaches that focus either on deliberative or on a reactive. It promoted the metaphor of the ecosystem with rich interactions between different species to maintain and exploit the information landscape. The idea defended here is to complement the emergent intelligence metaphor and the intelligent agent society metaphor reusing the same ecological chains that occur in nature *i.e.* more organized or intelligent species relying and/or farming others (e.g., ants and aphids) and vice-versa. In the future, intelligent agents could use heuristics, rules or norms they have been explicitly given or they have derived, on how to use swarm intelligence in achieving their goals. This article showed several examples of interactions between reactive and deliberative agents that were successfully used to implement semantic Web applications.

In all the examples described previously, simple mobile agent propagation relied on classic generic protocols (e.g. FIPA-Request) avoiding the development of specialized protocols or speech-acts for every task; in a way, reactive agents encapsulate ad-hoc protocols. Their natural heterogeneity makes them an excellent tool to address the natural heterogeneity of tasks to be propagated. Many open questions remain to be addressed, such as proper ways to manage and monitor a population of reactive agents, but they do not challenge the interest of the approach itself.

~~~

Acknowledgements To my colleagues in ACACIA and in the Mobile Commerce Laboratory of CMU. To the IST Program (CoMMA and SWAP projects), to the DAML initiative, to the Air Force Research Laboratory (contract F30602-02-2-0035), to the Defense Advanced Research Project Agency (contract F30602-98-2-0135). The US Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. This work was also in part supported by grants from IBM, HP, Symbol, Boeing and Fujitsu. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

References

[1] L. Floridi, "Information Ethics: On the Theoretical Foundations of Computer Ethics", *Ethics and Information Technology*, 1999 1 (1), pp. 37-56,

[2] T. Berners-Lee, J. Hendler, O. Lassila, "The Semantic Web", In *Scientific American*, May 2001, p35-43

[3] World Wide Web *Recommendations,* www.w3.org, June 2003

[4] M. Klusch, "Information Agent Technology for the Internet: A Survey", *Journal on Data and Knowledge Engineering*, Special Issue on Intelligent Information Integration, D. Fensel (Ed.), Vol. 36(3), Elsevier Science, 2001

[5] H. V. Parunak, S. Brueckner, J. Sauter, "ERIM's Approach to Fine-Grained Agents", NASA Workshop on *Radical Agent Concepts*, Greenbelt, MD, USA, Jan. 2002.

[6] E. Bonabeau, M. Dorigo, G. Theraulaz, *Swarm Intelligence*, Oxford University Press, 1999.

[7] O. Babaoglu, H. Meling, A. Montresor, "Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems", in *Proceedings 22nd International Conference on Distributed Computing Systems* (ICDCS '02), Vienna, July 2002.

[8] K. P. Sycara, "Multiagent Systems", *AI Magazine*, 19(2): Summer 1998, 79-92

[9] P.-P. Grassé, "La Reconstruction du nid et les Coordinations Inter-Individuelles chez Bellicositermes Natalensis et Cubitermes sp. La théorie de la Stigmergie: Essai d'interprétation du Comportement des Termites Constructeurs." *Insectes Sociaux*, 6:41-84, 1959.

[10] A. Newell, "The knowledge level", *Artificial Intelligence* 18 (1982) 87–127.

[11] G. J. Bex, S. Maneth, F. Neven., "A formal model for an expressive fragment of XSLT", *1st International Conference on Computational Logic*, pp. 1137-1151, LNAI 1861, Springer, 2000

[12] F. Gandon, "Distributed Artificial Intelligence and Knowledge Management: ontologies and multi-agent systems for a corporate semantic web", *PhD dissertation*, INRIA and University of Nice - Sophia Antipolis, 7th of November 2002. http://www-2.cs.cmu.edu/People/fgandon/research/ or http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/

[13] F. Gandon, N. Sadeh, "A Semantic e-Wallet to Reconcile Privacy and Context Awareness", to appear in proceedings of ISWC 20-23 October 2003, Sanibel Island, Florida

[14] T. Cao, F. Gandon, "Integrating external sources in a corporate semantic web managed by a multi-agent system" *AAAI Spring Symposium* on Agent-Mediated Knowledge Management, March 24-26, 2003, Stanford pp. 123-130

[15] PubMed, 2003, http://www.ncbi.nlm.nih.gov/entrez/

[18] F. Bellifemine, A. Poggi, G. Rimassa, "Developing multi agent systems with a FIPA-compliant agent framework", *Software Practice & Experience*, (2001) 31:103-128

[19] O. Corby, C. Faron-Zucker, "Corese: A Corporate Semantic Web Engine", *Workshop on Real World RDF and Semantic Web Applications*, 11th International W3 Conference 2002 Hawai

[20] P. Horn, "Autonomic Computing: IBM's Perspective on the State of Information Technology". http://www.research.ibm.com/autonomic/manifesto/autonomic_computing.pdf July, 2003.
~~~