

# **Metrics for Intelligence: the Perspective from Software Agents**

## ***PRELIMINARY NOTES***

**Workshop on Performance Metrics for Intelligent Systems**  
**National Institute of Standards and Technology**

**Line Pouchard**

Collaborative Technologies Research Center  
Computer Science and Mathematics Division  
Oak Ridge National Laboratory  
Oak Ridge, TN 37831-6414  
pouchardlc@ornl.gov

Each scientific development that claims to provide a “new way” for approaching existing problems needs proper (i.e. formal and quantifiable) evaluation methods and consensus-based criteria for measuring the validity of its claims. Taken together, these methods and criteria constitute the metrics by which new developments are being measured against their claims. Various claims have been made in the literature for the technology of intelligent software agents. Such claims include a new approach to programming providing a breakthrough comparable to the one achieved through object-oriented methods; an approach to programming that is more readily understood by non-programmers; an approach that lowers the costs of software inter-operability.

Software agents need proper metrics if the technology is to fulfill its promises and make a lasting impact. One characteristic distinguishing software agents from software developed with object-oriented and procedural methodologies is the anthropomorphic characteristics that agents exhibit. Various taxonomies for software agents currently exist [1, 2, 3]. Agents typically present one or several of the following characteristics:

- Pro-activeness and goal-orientation
- Reactiveness (reactive agents)
- Autonomy (rational agents, and others)
- Mobility (mobile agents)
- Learning and reasoning ability (deliberative agents, and others)
- Social ability: communication and cooperation (multi-agent systems)

An agent is considered intelligent if it can learn from its environment and modify its behaviors and goals to respond to environmental constraints that were uncertain and unforeseen at the time of development. Agents are thus particularly adapted to model environments where software components act autonomously on users’ behalf and problem-solving environments where parameters of computation dynamically change during processing. The ability to learn for an agent is coupled with the ability to perform resource and knowledge discovery. This action may take the form of querying and updating knowledge-based systems. Knowledge discovery and interpretation bring latency to the agent and may impair the achievement of its overall goals. For instance, reactive agents that need a quick response time may not embody much learning and reasoning because the overhead renders the agent useless.

Software agents present one or some capabilities that are affected by the choice of specific components described in the Tools of Intelligence (see White paper). For instance, searching for a required object within a scene is one area where software agents have successfully been implemented. If you take the “scene” to be an information space like the Internet, information-gathering and retrieval agents display this capability and have been successful at performing the task. Deliberative agents such as Belief-Desire-Intention (BDI) agents exhibit the capability of remembering scenes and experiences as their Beliefs are based on this capability. These agents are also able to interpret and respond to unforeseen situations.

Agents’ ability to autonomously execute processes on remote systems, given the appropriate permissions, is also a characteristic some intelligent systems (but not all) need to efficiently and effectively perform. This requires proper measures. This characteristic, known as mobility, has very different meaning for physical agents.

Mobility requires intelligence for software agents because true mobility requires resource discovery. For those agents designed as mobile agents the degree of mobility can constitute a measure of its intelligence. Mobile agents travel over networks such as the Internet and execute processes on remote platforms. Mobile agents may start execute a process on a particular machine, be unexpectedly interrupted, travel to another available platform, and continue the execution of the process from where it was interrupted. Such a mobile agent needs intelligence to interrupt and restart its execution autonomously without resetting, and for determining which resources to use in a networked environment. Network agents used for telecommunication applications (such as testing the reliability of a network) exemplify these types of agents.

Social intelligence needs to be measured in multi-agent systems. The degree of social interaction and the agents’ ability to exhibit social behavior constitute an important criterion for multi-agent systems. Not all agent-based systems need to exhibit this characteristic (mobile agents may never need to talk to each other for instance). The type of social interaction between agents conditions knowledge acquisition and interpretation. The social model affects the individual pursuit of goals and may ultimately affect the survival of the system [4]. When one considers a multi-agent systems, there are at least two models. Both types of multi-agent systems, collaborative and cooperative, display the characteristics of open systems.

- Model 1: Each individual agent’s goal is subservient to an over-arching goal of the system. We have a cooperative system, where agents agree not to pursue goals detrimental to each other and the whole system, even if these “careless” goals are in accordance with the individual agent’s goal.
- Model 2: Each agent acts on its own behalf without recognizing a higher agent-entity with the ability to regulate its goals (there is still a need for a kind of supervisor agent that regulates communication). We have a collaborative system. This is the case for so-called rational agents, used especially in e-commerce, where agents act in a market-like environment, with the ability to bid for money on the goods and services each offers.

Agent-communication languages should theoretically let heterogeneous agents communicate, but none currently do [5]. A significant part of the inter-operability issue is the lack of a shared content language and ontology. An ontology expresses, for a particular domain, the set of terms, entities, objects, classes

and the relationships between them with formal definitions and axioms that constraint the interpretation of these terms [6]. These definitions and axioms are written in a variety of logical languages (e.g. KIF [7]), and provide a formal theoretical basis to domain taxonomy. They can serve to automatically infer translation engines between software applications. By making explicit the implicit definitions and relations of classes, objects, and entities, ontologies also contribute to knowledge sharing and re-use across systems. The use of ontologies in agent-based systems is proposed as a criterion for the metrics of intelligent software agents. The degree of completeness and consistency of ontologies can be formally proven and provide a quantifiable criterion.

Ontologies constitute an important criterion for the metrics of intelligent software agents, in particular for agents exhibiting the social abilities of communication and cooperation. Software agents require the use of or a translation to a shared terminology and syntax in order to efficiently and effectively inter-operate. Agent-communication languages such as KQML meet the challenges of inter-operability with mitigated success [8]. Agent communication languages specify the possible use of ontologies in their syntax but do not require it. FIPA ACL proposes an ontology service as a normative specification [9].

In conclusion, software agents exist either as standalone or in social systems. Agents are made of components, and an agent-oriented architecture typically includes the agent application as well as an environment in which agents execute. They may execute on a single machine, on several machines connected locally or by wide-area network. These agents need a degree of mobility. They may be developed by different developers on different platforms, and therefore need a common communication language including protocol and ontologies (see [10] for an assessment of the state-of-the-art in this area). In addition, since agents may exhibit any combination of the characteristics above, some taxonomies of agents prefer a classification based on the domains in which software agents have been successfully implemented [11], rather than on their inherent characteristics.

Software agents also exist as whole, where an agent-based system is made of the agent and the underlying environment. The environment may include the knowledge repositories and ontologies which are key to the agents' degree of intelligence. For this reason, the mind/body dichotomy, and the proposition to measure the intelligence of the system based on the intelligence of the mind (controller), do not hold for agent based systems.

In addition to characteristics applicable to Constructed Systems with Autonomy, the metrics of intelligence for software agents need to include the following (not all these characteristics need apply for the same system):

- be domain-specific
- measure the degree of mobility
- present an agent communication language
- refer to ontologies.

## References

1. Brenner, Walter; Zarnekow, Rudiger, and Wittig, Hartmut. *Intelligent Software Agents. Foundations and Applications*. Berlin: Springer Verlag; 1998; pp. 37-41.
2. Nwana, H. et. Al. "What is an agent?" Available at:  
<http://www.labs.bt.com/projects/agents/publish/papers/review2.htm#agent>
3. Wooldridge, M. Intelligent Agents. In *Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence*. Weiss, Gehrard, ed. Cambridge, Mass.: MIT Press; 1999; pp.27-73.
4. Huhns, Michael N. and Stephens, Larry. Multiagent Systems and Societies of Agents. In *Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence*. Weiss, Gehrard, ed. Cambridge, Mass.: MIT Press; 1999; pp. 79-120.
5. Singh, Munindar P. Agent communications languages: rethinking the principles. *IEEE Computer*. 1998; 31(12):40-47.
6. Gómez-Pérez, Asuncion. Knowledge Sharing and Reuse. In *The Handbook of Expert Systems*. Boca Raton, FL: CRC Press; 1998; pp. 10/1-10/36.
7. Knowledge Interchange Format. Available at <http://logic.stanford.edu/kif/dpans.html>.
8. Labrou, Yannis; Finin, Tim, and Peng, Yun. Agent communication languages: the current landscape. *IEEE Intelligent Systems & Their Applications*. 1999; 14(2):45-52.
9. Foundation for Physical Intelligent Agents (FIPA). Available at <http://drogo.cse.it/fipa> and <http://www.fipa.org>.
10. Wooldridge, M. and Jennings, N.R. Applications of Intelligent Agents. In *Agent Technology: Foundations, Applications, and Markets*. Berlin: Springer Verlag, 1998; pp. 3-28.
11. Nwana, Hyacinth S. and Ndumu, Divine T. A perspective on software agents research. *The Knowledge Engineering Review*. 1999; 14(2):125-142.

The submitted manuscript has been authored by a contractor of the U.S. Government under Contract No. DE-AC05-00OR22725. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.