SPECIFYING AND VALIDATING THE AGENT PERFORMANCE EVALUATION METHODOLOGY: THE SYMBIOSIS USE CASE

4 AUTHORS, INCLUDING:

Fani A. Tzima
Aristotle University of Thessaloniki
16 PUBLICATIONS **46** CITATIONS
SEE PROFILE

Andreas Symeonidis
Aristotle University of Thessaloniki
79 PUBLICATIONS **336** CITATIONS
SEE PROFILE

Pericles A. Mitkas
Aristotle University of Thessaloniki
250 PUBLICATIONS **1,372** CITATIONS
SEE PROFILE

SPECIFYING AND VALIDATING THE AGENT PERFORMANCE EVALUATION METHODOLOGY: THE SYMBIOSIS USE CASE*

Christos Dimou, Fani A. Tzima, Andreas L. Symeonidis, and Pericles. A. Mitkas *
Dept. of Electrical and Computer Engineering,
Aristotle University of Thessaloniki,
GR-541 24, Thessaloniki, Greece

ABSTRACT

Despite the plethora of frameworks and tools for developing agent systems, there is a remarkable lack of generalized methodologies for assessing their performance, while adequately addressing the unpredictable and complex nature of intelligent agents. In this paper, we present a generic methodology for evaluating agent performance, the Agent Performance Evaluation (APE) methodology that consists of representation tools, guidelines and techniques for organizing and using metrics, measurements and aggregated characterizations of performance. The main element of APE is the Metrics Representation Tree, a generic structure that enables efficient manipulation of evaluation-specific information. A formal specification of the proposed methodology is provided and its applicability is demonstrated through Symbiosis, an existing multi-agent system to be used as a testbed.

KEYWORDS

evaluation methodology, agent performance, formal specification, metrics representation, Z notation

1. INTRODUCTION

During the last years, the interest in agent technology has evolved from the "agent-hype", earlier in this decade, to its adoption for a more narrow, yet better defined, range of applications that exploit certain engineering characteristics of agents, including adaptation to dynamic environments, mobility and negotiation capabilities. Still, although many interesting findings have been produced in the research sphere (including incorporation of cutting-edge AI techniques into agent bodies), very few real-world applications actually use agents in practice, a fact that underlines the reluctance of software practitioners to embrace agent systems.

We argue that one reason for this is that, despite the plethora of engineering tools and methodologies for agents, there exists no general, standardized performance evaluation methodology that enables developers to formally validate and quantify benefits and drawbacks that agents and MAS may exhibit. Given the absence of such a general evaluation methodology, researchers in the agent community currently devise their own metrics and methods for special purpose evaluation. The corresponding findings are of qualitative nature, thus introducing some degree of subjectivity, as well as practically impossible to reproduce, due to not having followed a well defined methodological path.

In the literature of Intelligent Systems evaluation (agents and MAS being a special case of this field), two complementary approaches to the IS evaluation problem can be identified [1]: (i) the bottom-up and (ii) the top-down. The former advocates the definition of formal constructs and languages that will enable the definition of the appropriate terms and scope of IS (e.g. [2]). Evaluation will thereafter be gradually built upon these formal foundations. The top-down approach (e.g. [3]), on the other hand, supports experiences

* This paper is part of the 03ED735 research project, implemented within the framework of the "Reinforcement Programme of Human Research Manpower" (PENED) and co-financed by National and Community Funds (25% from the Greek Ministry of Development - General Secretariat of Research and Technology and 75% from E.U.-European Social Funding).

from different ad-hoc evaluation attempts may be generalized into a concise domain-independent methodology, that will be established at the time that IS reach a sufficient level of maturity

In this paper, we present a complete methodology, top-down for evaluating the performance of agents and MAS using the Symbiosis [9] test case for validation purposes. The Agent Performance Evaluation (APE) methodology's main objectives are (i) to provide structured representation tools for organizing and using evaluation-specific knowledge; (ii) to standardize various steps of the evaluation process, including metrics, measurements and their aggregation; and (iii) to incorporate higher level (qualitative) concepts into quantitative evaluation processes. In this direction, we provide a formal specification of the presented methodology, using Z notation [4]. APE is then defined as a "plug-in" for any system instantiation, in the sense that the definitions provided are abstract enough to cover a wide variety of systems and application domains.

The remainder of this paper is structured as follows: Section 2 provides a formal specification of the APE methodology; Section 3 examines the applicability of APE in a specific application domain and Section 4 concludes the paper with a summary of our contributions and future research directions.

2. THE APE METHODOLOGY

2.1 Overview

The Agent Performance Evaluation (APE) methodology, a set of representation tools and guidelines for evaluators in the field of agent computing. APE is part of a more general evaluation framework that also addresses the use of automated tools for conducting online collection of measurements during the runtime of a MAS (MEANDER in [5]). In this paper, we focus on the details of APE and examine its contribution to each one of the three following aspects of evaluation.

***Metrics*:** The Metrics Representation Tree (MRT), whose general structure is depicted in Figure 1, is a representation tool that organizes metrics, from specific to general aspects of performance evaluation. The bottom-most level of the MRT consists of *Simple Metrics* (SM) that are directly measurable. SMs are combined to form *Composite Metrics* (CM) that are not directly measurable, representing higher-level performance concepts. Moving upwards in the tree, CMs as well as SMs are further combined to form more CMs, until the root of the tree which corresponds to the overall *System Performance*

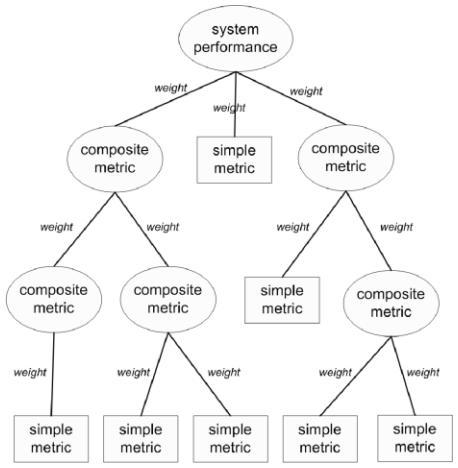

Figure 1: The Metrics Representation Tree

***Measurement*:** Having selected the measurement method (e.g. with respect to the nature of experiments [6]), one must thoroughly provide an experimental design prototype and a data collection procedure. Since measurement techniques heavily depend on the application at hand, APE does not provide details on *how* measurements are conducted. Instead, it emphasizes the automation of measurement collection, an issue

addressed within the MEANDER framework, a specific implementation of automated evaluation that follows the APE principles [6].

***Aggregation*:** Following the collection of measurement values and the construction of metric-measurement pairs, the problem of aggregation arises, in order to summarize experimental findings into a single characterization of performance - either of a single module, or the system as a whole. In the case of the MRT of the proposed methodology, after having collected the measurements, the user traverses the tree in a bottom-up manner: starting from the specific metrics view, he/she proceeds upwards and, at each view, applies aggregation techniques to provide single characterizations of parent nodes. The MRT aggregation techniques one can apply range from simple average operators to complex multi-criteria decision-making [7].

2.2 Specification of APE

Agents and MAS specification: The basis upon which we have developed our methodology is a formal, abstract specification of agent environments (*System*). The definition of basic terms and notions is imperative, in order to both unambiguously use terms and implicitly define the scope of applicability of the proposed methodology. Thus, in this paper, we employ several basic definitions of agent-related concepts found in the work of d'Inverno and Luck [8], wherein a concise and sufficiently abstract definition of agents and MAS is provided. The specification in [8] begins with an axiomatic definition of *Attributes*, *Motivations*, *Actions* and *Goals*, and gradually proceeds to more complex terms, such as *Agent* and *MAS*. The reader is encouraged to refer to the original publications for the formal specifications of the basic terms described below:

1. In the context of MAS, a System consists of a set of *Entities* (agents or static objects) and a set of attributes (*sysAttrs*), describing its various properties.
2. An *Entity* is an abstraction for any possible participant in the *System* that may have a set of possible actions, goals and other attributes (*enAttrs*).
3. An *Agent* is a special case of an *Entity*, required to have non-zero goals.
4. A *View* can be perceived by an *Agent* at a given time of runtime execution and includes a non-empty subset of *System* attributes.
5. A *MAS* is a conglomeration of entities, hierarchically ordered as objects, agents and autonomous agents. Several different types of agents may participate in a *MAS* (namely *autonomous agents* and *server agents*, both specified in [10]), but it is required that any *MAS* consists of at least two agents, at least one of which is an autonomous agent. Finally, it is required that at least two agents share a minimum of one goal.

Evaluation: The central concept of APE is the *metric*, a standard that defines measurable attributes of entities, their units and their scopes. Metrics are the essential building blocks of any evaluation process, as they allow for the establishment of specific goals for improvement. The metric selection process is, thus, the answer to the question: "*Which system aspects are representative of its performance?*".

In general, the term metric may refer to (i) global performance aspects of the system that are known to all agents and/or measurable by central observers or non-agent entities (such as logfiles and databases); or (ii) agent performance attributes that are only accessible by the agent in question. Moreover, a *Metric* may either be a *SimpleMetric* or a *CompositeMetric*, as defined in: $Metric \equiv SimpleMetric \vee CompositeMetric$

$$
\begin{array}{|l}
\hline
SimpleMetric \\
val : Value \\
\hline
val.type = \{simple\} \\
val.value \in enAttrs \vee value \in sysAttrs \\
\hline
\end{array}
$$

The *SimpleMetric* schema introduces the *value* member, which represents the actual measured value for this metric. The measurability requirement for a *SimpleMetric* is satisfied by ensuring that the *value* property is either an entity (*enAttrs*) or a system (*sysAttrs*) attribute. In the above schema, *Value* is defined as:

$$Value \equiv [value : Attribute; type : Attribute; scale : Attribute]$$

CompositeMetric

$$
\begin{aligned}
&System \\
&val : Value \\
&children : \mathbb{P}\, Metric \\
&weights : \mathbb{P}\, Weight \\
&weightedChildren : \mathbb{P}\, Metric \nrightarrow \mathbb{P}\, Weight \\
\hline
&val.value \notin sysAttrs \land value \notin enAttrs \\
&dom(weights) = children \\
&\sharp children > 0 \\
&\sharp weights = \sharp children \\
&weights = weightedChildren\ \ children \\
&type = \{composite\}
\end{aligned}
$$

A *CompositeMetric* is a higher-level metric that is not directly measurable (note the requirement in the constraints section that value is not a member of either *sysAttrs* or *enAttrs*); instead it is constructed from a number of children metrics. The contribution of each child to the *CompositeMetric* is signified by a *Weight*, defined as *Weight* ≡ *{value : R}*. A one-to-one function, named *weightedChildren*, maps children to weights, while the actual aggregation function for a *CompositeMetric* is later defined, in the *Aggregate* schema.

Finally, the *MRT* schema contains metrics that comprise both a set of *SimpleMetric* and *CompositeMetric* instances. The status member takes values in the set *S = {EMPTY, PARTIALLY_BOUND, BOUND, AGGREGATED}*, signifying an *MRT* that has zero, some, all Simple Metrics values and all Composite Metrics values, respectively. The last two constraints ensure the tree structure of the *MRT*. We require that: (i) all *SimpleMetrics* are children of some *CompositeMetrics* and (ii) there exists a single root node that has no parents and all other metrics are direct or indirect children of it.

MRT

$$
\begin{aligned}
&metrics : \mathbb{P}\, Metric \\
&sMetrics : \mathbb{P}\, SimpleMetric \\
&cMetrics : \mathbb{P}\, CompositeMetric \\
&status : Attribute \\
\hline
&cMetrics \cup sMetrics = metrics \\
&status \in \{EMPTY, PARTIALLY\_BINDED, BINDED, AGGREGATED\} \\
&\forall\, s \in sMetrics \mid (\mu\, c : CompositeMetric \bullet s \in c.children) \\
&\mu\, root \in cMetrics \mid (\forall\, cp \in cMetrics \bullet root \notin cp.children) \land \\
&\quad [m \in \{metrics \setminus root\} \mid (\exists\, p_1 \in cMetrics \bullet m \in p_1.children) \land \\
&\quad (\exists\, p_2 \in metrics \bullet p_2 \in root.children)]
\end{aligned}
$$

Evaluation Operations: Several operations are required for the association of the APE methodology to the actual experiments of a runtime MAS, in order to: (i) collect the measured values for metrics within the context of an experiment (ii) integrate these values in the MRT and (iii) aggregate the MRT. We start with Measurement that, having selected the appropriate metrics, is the next fundamental methodological step that systematically assigns specific values to these metrics. Typical measurement methods consist of experimental design and data collection. A measurement method is, thus, the answer to the question "How should I perform the experimental evaluation?". The *MeasurementCollector* schema is simply a declaration of system entities authorized for collecting measurements and binding them to the *MRT*.

MeasurementCollector

$$
\begin{aligned}
&System \\
&MAS \lor Agent \lor StaticObject \\
\hline
&\forall\, e : Entity \mid e \in MAS.entities \bullet e \subseteq System.entities \\
&Agent \subseteq System.entities \\
&StaticObject \subseteq System.entities
\end{aligned}
$$

Then, we define the *Experiment* schema, from the APE perspective. Any experiment involves the actual system, a set of measurement collecting entities and a set of experimental parameters (such as initiation and termination conditions, number of iterations, number of participating entities).

Experiment

$system : System$
$mrt : MRT$
$measurmentCollectors : \mathbb{P}\ MeasurementCollector$
$params : \mathbb{P}\ Attribute$

$measurmentCollectors \in system.entities$
$\forall s \in mrt.sMetrics \bullet s.val.value \in system.sysAttrs$
$\forall measurementCollectors.entities \subseteq system.entities$
$\forall params \subseteq system.sysAttrs$

The relationship between the system, the APE methodology and the experiment, is satisfied by the fact that all *SimpleMetrics* in the *MRT* refer to actual system attributes, and all measuring entities participate in the system runtime, while all experimental parameters are valid within the context of the system.

The next operation, *BindValue*, associates measured values with the appropriate *SimpleMetrics*. The schema's input is a measured value that modifies a certain *SimpleMetric* (*ΔSimpleMetric*) by applying the bind function to it. The constraints ensure that values share the same type with metrics, in order for the correspondence of metric and measurement to be correct.

BindValue

$\Delta SimpleMetric$
$v? : Value$

$v?.type = SimpleMetric.value.type$
$SimpleMetric'.value = v?$

The above schema binds a value to a single metric. However, in order to define the impact of this change to the entire MRT, we employ the promotion technique, defined by Z notation. We define the schema *PromoteBind*:

PromoteBind

ΔMRT
$\Delta SimpleMetric$
$s? : SimpleMetric$

$s? \in MRT.sMetrics$
$\theta SimpleMetric = s?.value$
$MRT'.sMetrics = MRT.sMetrics \oplus \{\theta SimpleMetric'.value\}$

For each measured value, we obtain a schema that describes the operation on the entire MRT by ensuring that $BindValues \equiv \exists \Delta SimpleMetric \bullet BindValue \wedge PromoteBind$

Another critical operation on Composite Metrics is *Aggregation*, which groups and combines collected measurements, possibly by the use of weights of importance, in order to conclude to atomic characterizations of the evaluated system. Aggregation is the answer to the question: "*What is the outcome of the evaluation procedure?*". The aggregation operation calculates the value of a *CompositeMetric* based on an appropriate function on the values of its children.

Aggregate

$\Delta CompositeMetric$
$calculateCompositeMetric : \mathbb{P}(Value \times Weight) \rightarrow Value$

$c'.value = calculateCompositeMetric\ (children.value \times weights)$

Similar to the *BindValue* schema, *Aggregate* is also promoted and, for each measured value, a schema that describes the operation on the entire MRT is defined as:

$$AggregateMRT \equiv \exists \Delta CompositeMetric \bullet Aggregate \wedge PromoteAggregate$$

Finally, we provide the *MeasureValue* schema used by instances of *MeasurementCollector*:

$$MeasureValue \equiv value! : Value \vee f : PAttr \rightarrow Value$$

In conclusion, the experimental phase of evaluation consists of the following successive steps: (i) the parameters of the experiment and the corresponding MRT are defined; (ii) the system enters the runtime phase; (iii) the measuring entities measure the performance values; (iv) the collected values are bound to the MRT; and (v) the MRT is aggregated.

The Z-statement *Experiment ; InitSystem ; MeasureValues >> BindValues >> AggregateMRT* resumes the above steps, by including the corresponding schemas. In the previous statement, *InitSystem* is an operation that initializes the members of the *System* schema, the *";"* operator denotes a succession of schema inclusions or operations, while the >> operator denotes a piped execution of operations.

3. DEMONSTRATION

In this section, the applicability of APE is demonstrated through an already implemented multi-agent simulation environment, namely Symbiosis [9], that aims to study various aspects of learning capabilities of agent societies. Symbiosis simulates a virtual ecosystem that hosts two self-organizing, combating species: preys, which are herbivorous and consume resources from the environment, and predators, which are carnivorous and consume preys. All agents (animats) live and evolve in this shared environment, they are self-maintaining and engage in a series of vital activities, with the ultimate goals of survival and reproduction. Moreover, all their decisions are determined by an agent learning mechanism modeled using classifier systems and genetic algorithms [10].

3.1 The Symbiosis MRT

We proceed by defining the appropriate metrics in the MRT structure. This process, currently undertaken by a domain expert, requires the identification of metrics and their weighted associations (Figure 2). Table 1 lists the Simple Metrics, as introduced and applied in experiments in [9].

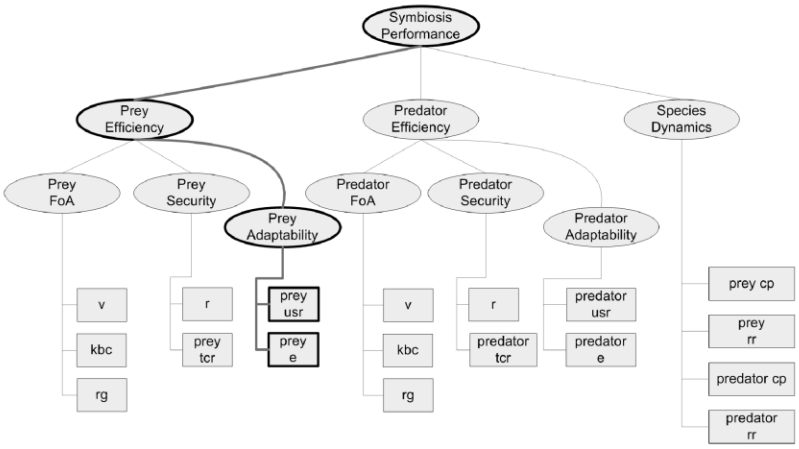

Figure 2: The Symbiosis MRT

Composite Metrics in the Symbiosis MRT are briefly described below:

1. *Adaptability* is the ability of animats to learn and adapt in a changing environment. It is defined as (i) proportional to the animat's *effectiveness* that encompasses the efficient use of its energy and environmental resources; and (ii) inversely proportional to the unknown situations an animat encounters.

$$PreyAdaptability : CompositeMetric \mid$$
$$\theta CompositeMetric = \triangleleft children \rightsquigarrow \{prey\_e_{NET}, prey\_usr\} \triangleright$$
$$PredatorAdaptability : CompositeMetric \mid$$
$$\theta CompositeMetric = \triangleleft children \rightsquigarrow \{predator\_e, predator\_usr\} \triangleright$$
$$where \quad prey\_e_{NET};\ prey\_usr;\ predator\_e;\ predator\_usr : SimpleMetric$$

Table 1. Simple Metrics (SMs) for (a) Agent and (b) Environmental Attributes

Agent Attributes	Abbr
resource consumption rate	rcr
trap collision rate	tcr
unknown situation rate	usr
reproduction rate	rr
knowledge base completeness	kbc
rule generality	rg
Effectiveness	e
net effectiveness	e_{NET}

(a)

Environmental Attributes	Abbr
epoch	N
resource availability	a
Environmental variety	v
Environmental reliability	r
Current population	cp

(b)

2. *Freedom of action (FoA)* represents the behavioral variety of animats with respect to a changing environment. An animat with a large degree of *FoA* would essentially have a large and reliable knowledge base of rules (*kbc*) that indicate the optimal action in a given environment. Ideally, an efficient knowledge base would have a large number of generalized rules (*rg*).
3. *Security* measures the ability of animats to move and act safely within the environment. It is evident that a high trap collision rate (*tcr*) signifies an inefficient action selection mechanism. Moreover, *Security* depends on the environmental reliability (*r*), since a highly unreliable environment would be a more challenging ground for animats to move and act safely.
4. *SpeciesDynamics* is an environmental metric that measures the "balance" between the two combating species and, therefore, takes into account population (*cp*) and reproduction rate (*rr*) of both species, in order to measure the effect of one group as a whole to the other.

Based on the above definitions, a sample instantiation of PreyAdaptability would have the form:

$$
\begin{aligned}
&PreyAdaptability : CompositeMetric \mid \theta CompositeMetric = \\
&\quad \lhd children \rightsquigarrow \{prey\_e_{NET}, prey\_usr\}, \\
&\quad weightedChildren \rightsquigarrow \{prey\_e_{NET} \rightarrow 0.6, prey\_usr \rightarrow 0.4\}\}, \\
&\quad value \rightsquigarrow Aggregate.caclulateCompositeMetric\ \ children\ \ weights \rhd \qquad (1) \\
&where \quad usr : SimpleMetric \mid \theta SimpleMetric = \\
&\quad \lhd aAttr \rightsquigarrow \{prey\_net\_efficiency\}, value \rightsquigarrow 0.151 \rhd \\
&\quad prey\_e_{NET} : SimpleMetric \mid \theta SimpleMetric = \\
&\quad \lhd aAttr \rightsquigarrow \{prey\_unknown\_situation\_rate\}, value \rightsquigarrow 0.942 \rhd
\end{aligned}
$$

3.2 Experiments

For demonstration purposes, we have repeated the experiments for Symbiosis, described in [9]. The goal of this section is to show that the proposed methodology can be applied to already existing systems, adding quantitative characteristics to higher-level evaluation concepts.

Tzima et al [9] describe a set of experiments that produce measurements for the metrics of Table 1. The authors proceed by qualitatively discussing the higher-level evaluation concepts, such as prey adaptability. By using the MRT of Figure 2, we are able to introduce actual functions for measuring such concepts. More specifically, we organize two sets of experiments, E_{A1} and E_{A2}, aiming to assess the effect of the genetic rule creation mechanism on animat performance. This being equivalent to measuring the adaptability of preys, we select the highlighted path of the MRT of Figure 2, which contains the set simple metrics *sm = {prey_usr, prey_e}* and the set of composite metrics *cm = {preyAdaptability, preyEfficiency, SymbiosisPerformance}*, and provide the instantiation in Z notation of the related schemas:

$$
\begin{aligned}
&E_{A1} : Experiment \mid \theta Experiment = \lhd system \rightsquigarrow \{Symbiosis\}, mrt \rightsquigarrow \{SyMRT\}, \\
&\quad measurementCollectors \rightsquigarrow \{Preys\}, params \rightsquigarrow ParamsSet1 \rhd \\
&E_{A2} : Experiment \mid \theta Experiment = E_{A1} \oplus \lhd params \rightsquigarrow ParamsSet2 \rhd
\end{aligned}
$$

where *ParamsSet1* and *ParamsSet2* are the set of parameters as defined in [9]. It must be noted that, while the global parameters are the same in both experiments, in E_{A1} both species utilize the learning mechanism, whereas in E_{A2} only preys are capable of learning through their actions.

The measurement and binding procedures result into two instances of the MRT, one for each experiment. We apply aggregation on the selected path of metrics for both MRTs, in a bottom-up fashion, using simple weighted sum as the aggregation function. The instantiation of *AggregateMRT* schema results into Composite Metrics of the form of Eq. 1 and, finally, into two aggregated MRTs, one for each experiment. Finally, comparing the aggregation results of $MRT(E_{A1})$ and $MRT(E_{A2})$, we observe that indeed *preyAdaptability*, and therefore *preyEfficiency* is higher in experiment E_{A2}, reaching the same conclusion as in [9], that preys adapt better when both species employ learning mechanisms.

4. DISCUSSION

In this paper, we have presented a generic evaluation methodology for agents and MAS that provides a coherent framework of tools and guidelines for organizing and using evaluation-related information. More specifically, the Metric Representation Tree (MRT) provides a structure for defining and quantifying higher-level evaluation concepts (composite metrics) and relating them to existing, directly measurable performance aspects (simple metrics). This way, qualitative features can now be integrated in the evaluation process in a quantitative manner. During the subsequent phases of measurement and aggregation, measured values are bound to the MRT and composed to produce performance characterizations, respectively. The evaluator is, thereafter, able to focus on any sub-tree of the MRT, in order to isolate aspects of system performance or examine the entire MRT for obtaining a single characterization of the system performance.

The definition of the metrics and weights in the MRT is currently undertaken by domain experts, introducing some degree of subjectivity. There are several techniques for automating this procedure. For example in [7], we propose a training method for the MRT that assesses the validity and certainty of the selected metrics and weights, based on historical performance data. Towards this direction, we envision a community of experts for each application domain that would define MRTs to be continuously refined by using historical performance data on this domain. Thus, evaluators would have access to readily-available, domain-specific MRTs, being relieved from the burden of re-inventing metrics and therefore avoiding current ad-hoc evaluation practices.

REFERENCES

1. Madhavan, R., Messina, E.,2000. Measuring performance and intelligence (white paper). *In Messina, E., Mystel, A., eds.: Proc. of PerMIS00*. Gaithersburg MD, USA, pp. 65-80.
2. Zadeh, L.A., 2002. In quest of performance metrics for intelligent systems – a challenge that cannot be met with existing methods. *In Messina, E., Mystel, A., eds.: Proc.of PerMIS02.* Gaithersburg MD, USA, pp. 303–306
3. Albus, J., Messina, E.R., Evans, J.M., 2000. Measuring performance of systems with autonomy. *In Messina, E., Mystel, A., eds.: Proc. of PerMIS00*. Gaithersburg MD, USA, pp. 1–30
4. Spivey,M.J., 1989. *The Z notation: a reference manual*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA
5. Dimou, C., Symeonidis, A.L., Mitkas, P.A., 2009. An integrated infrastructure for monitoring and evaluating agent-based systems. *Journal of Expert Systems with Applications*, 36(4), pp. 7630-7643
6. Kitchenham, B.A., 1996. Evaluating software engineering methods and tool, part 2:selecting an appropriate evaluation method technical criteria. *SIGSOFT Softw. Eng. Notes*, 21(2), pp. 11–15
7. Dimou, C., Falelakis, M., Symeonidis, A.L., Delopoulos, A., Mitkas, P.A., 2008. Constructing optimal fuzzy metric trees for agent performance evaluation. *In: Proc. Of the 2008 International Conference on Intelligent Agent Technology.* Sydney, Australia, pp. 336–339.
8. D'Inverno, M., Luck, M., 2004. *Understanding Agent Systems*. SpringerVerlag
9. Tzima, F.A., Symeonidis, A.L., Mitkas, P.A., 2007: Symbiosis: Using predator-prey games as a test bed for studying competitive co-evolution. *In: Proc. of KIMAS07*. Boston MA, USA, pp. 115–120
10. Holland, J.H., 1992: *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA