Exploring Self-Explanation: The System Side

Johannes Fähndrich

DAI-Labor, Technische Universität Berlin
Faculty of Electrical Engineering and Computer Science
johannes.faehndrich@dai-labor.de

Following *Jennings* [4], the development of pervasive, complex and distributed computing systems is 'one of the most complex construction tasks humans undertake'. Challenges which directly evolve from the pervasive nature of applications are currently countered by development paradigms, such as Service Oriented Architectures (SOA) or Agent Oriented Software Engineering (AOSE), to name but a few. The basic idea of such development paradigms is to decrease the level of complexity of programming in dynamic environments. *Euzenat* and *Shvaiko* [3] refer to this principle as 'level of dynamics'. This level of dynamics increases with the amount of details that are left unspecified until the runtime of the application. Applications that we want to put our focus on are those which evolve over time and thus create additional heterogenousy problems. These problems are mainly due to the many different developers which are involved in the programming, each one making use of individually preferred technologies. However, it has been argued that systems with a high level of dynamic at runtime and a heterogeneous character, should account for so called self-* properties [8]. The self-* properties describe systems that are capable to adapt them selves to external and internal factors in a decentralized manner. In this work, it is our intention to foster one particular self-* property, namely the self-explanation property. Here, we discern between system-side self-explanation and human-side self-explanation and where system-side self-explanation refers to the ability of systems to describe themselves to other system components, human-side self-explanation refers to the ability of systems to describe themselves to human beings. As this work focus on the former characteristic, namely the system-side self-explanation, we intend to develop self-explaining descriptions, such as semantic and contextual information for agent functionalities. We will also account for the automated generation of knowledge tought reasoner. For this purpose we examine the algorithems of a third party reasoner.

When it comes to the design time of software applications, unspecified details always leave room for alternatives [6]. Each alternative requiring self-explanation capabilities in order help with the selection. We subdivide the problem of self-explanation into four subproblems. First of all, we need to answer what is an explanation? In this work explanations are created by adding semantic and contextual information to descriptions (at runtime), which will ease the inference of some reasoner observing the explanation. Some work was already done by *Overton* [7], yet, additional research is still required in order to make an explanation processable by agent planners or other reasoners.

Secondly, we need to know what kind of information should be contained in a self-explanation? This also implies the question on how much of a language needs to be known, to understand a description in this language [9]. Semantic information can be described by a *Natural Semantic Metalanguage* [2], reducing the amount of unexplained words needed to understand a description to approximately 62 Words.

Afterwards, we must accomplish a way to explain semantic information using such a metalanguage and enriching it with contextual information? In order to solve this task, we intend to analyze and extend state of the art methods and approaches which describe entities of dynamic and complex systems.

Subsequently, we want to answer the question in which way self-explanation supports an agent in its planing task? The last question abstracts for example the research done in the area of artificial intelligence, information extraction and planing, with the regards of extracting knowledge from information sources (here the given explanation). Such methods and tools were analyzed by *Drumond* and *Girardi* [1]. We will consider such algorithems of reasoning in our work.

For the theoretical evaluation we follow *Kaddoum et al.* [5], who proposed criteria for the evaluation of the adaptiveness of a system. As a practical evaluation idea of the self-explanation properties we may use a 'state of the art'-consumer of such descriptions like a agent planing systems and evaluate if the explanation produces better planing results (precision and recall values) then with traditional descriptions. To sum up, self-explanation can be used to connect distributed systems. The creation of such explanations and their use requires further research. The intention of this work is to foster self-explanation and an increased coupling of distributed systems. Hence, this work analyses the way components are described, how these descriptions are created and which methods of inference can be build upon them. The results of this work will help approaches using semantic descriptions, such as service matching, ontology alignment automated planning.

References

1. Drumond: A survey of ontology learning procedures. In: Ontologies and their Applications. vol. 427, pp. 1–12 (2008)
2. Durst: The natural semantic metalanguage approach to linguistic meaning. Theoretical Linguistics 29(3), 157–200 (2008)
3. Euzenat: Ontology Matching. Springer Berlin (2007)
4. Jennings: An agent-based approach for building complex software systems. Communications of the ACM 44(4), 35–41 (2001)
5. Kaddoum: Criteria for the evaluation of self-* systems. In: SEAMS '10. pp. 29–38. ACM, NY, USA (2010)
6. Müller-Schloer (ed.): Organic Computing - A Paradigm Shift for Complex Systems. Springer (2011)
7. Overton: Scientific explanation and computation. Explanation-aware Computing p. 41 (2011)
8. Schmeck: Organic computing-a new vision for distributed embedded systems. In: Object-Oriented Real-Time Distributed Computing. pp. 201–203. ISORC '05 (2005)
9. Steels: Modeling the cultural evolution of language. Physics of life reviews 8(4), 339–356 (2011)