# Autonomous Participation in Cloud Services

Josef Spillner, Christian Piechnick, Claas Wilke, Uwe Aßmann, Alexander Schill
*Faculty of Computer Science*
*Technische Universität Dresden*
*01062 Dresden, Germany*
*Email: {firstname.lastname}@tu-dresden.de*

***Abstract*—A Cyber-Physical System (CPS) is a combination of multiple physical devices connected and organised by a central controlling infrastructure using a feedback-loop mechanism. In order to increase the autonomy of CPS they must be connected to rich cloud services (e.g., social software and elastic resource services) enabling more sophisticated decision making. Traditionally, software agents are used in these systems to automate computational tasks by separating decision making from routine execution and hence letting the systems act autonomously. In this paper, the Advanced Autonomous Participation Scheme (AdAPtS) is proposed that can selectively cover autonomous computing on both the decision and the execution level, thus enabling classical cloud services embedded into CPS on a range from user-controlled devices to fully autonomous robots.**

## I. Introduction

Cyber-Physical Systems (CPS) are composed of physical devices (i.e., stationary, mobile and embedded systems, sensors) which are connected and organised by a logically centralised controlling infrastructure. On the lower control layers, the infrastructure is hierarchically segmented according to the device and subsystem network topology or functionality [1]. Since those systems depend on autonomic self-management [2], they must be able to *(1)* **sense** and *(2)* **analyse** the environment they are acting in, *(3)* **plan** appropriate actions based on these findings and *(4)* **perform** them. This kind of self-adaptivity is usually based on a closed feedback-loop mechanism (e.g. MAPE-K Loop) [3]. Additionally, these systems are often restricted in terms of communication, storage and network resources so that performing complex tasks without support from external services will be hard to impossible. Using cloud computing facilities to realise the central controlling infrastructure and discrete computation and storage tasks leads to numerous advantages (e.g. scalability, reliability or location transparency). Furthermore automated cloud computing and service orientation introduces the capability to dynamically publish and consume services that can be offered from and integrated into the lifecycle of Cyber-Physical Systems on demand. This way, the functionality of the overall system can be extended at runtime when it is necessary. Yet, even with capable cloud services available everywhere on the Internet today, finding and using them is still a labour-intensive and cumbersome effort, leading to a decreased quality of experience for humans and service lock-out for machines. Imagine, for instance, an autonomously acting robot running out of storage which requires a migration or extension of its storage capacity. Ideally, storage services could be integrated or reconfigured for the robot's application automatically, solely based on the most differential parameters such as cost or capacity to select most appropriate services. Instead, nowadays, several heterogeneous and manual web-based sign-up, confirmation and configuration steps need to be performed. Likewise, imagine a mobile phone which lets users take pictures and record videos without requiring an explicit sign-up to certain photo storage and retouching services. Instead, it just queries interactively for the permission to do so using the user's profile and the desired service properties, for instance, a free service within a certain country. This, too, is not possible with today's cloud integration and cloud service participation techniques.

In order to adaptively integrate cloud services into a CPS at runtime, software systems must be able to autonomously pass typical service consumption lifecycle steps: the selection, sign-up and registration, configuration, usage and authentication processes.

In this paper, we propose the *Advanced Autonomous Participation Scheme* (AdAPtS) for cloud services, which enables software agents to automatically select, sign up, log in and use cloud services. We apply the scheme to an example scenario, which shows the communication and interaction of human and artificial participants via social cloud services. Yet, the scheme is universally applicable to all service-oriented environments with sufficient descriptions and interfaces. As a result, the scheme leads to a decreased effort for human participants with interactive devices as well as already autonomously acting cyber-agents and robots.

The remainder of this paper is structured as follows: First in Sect. II, we provide the scope of cloud services and the automation potential for their lifecycle. Afterwards, we give a short example for a CPS application based of human and robot stakeholders interacting via cloud services in Sect. III and present the main characteristics of AdAPtS in Sect. IV. An automation framework for resource-constrained cloud service controllers with respect to CPS is described afterwards and concludes the article with an outlook in Sect. V.

## II. Background

### A. Cloud Service Model

Cloud computing is defined as the sum of the service modelling paradigm *Everything as a Service* (XaaS) and the service delivery paradigm *Utility Computing*, arising from the different levels of the Cloud Computing Service Model [4]. In the scope of this paper, cloud computing is treated rather broadly as a combination of Internet and web services (e.g., communication or social activity services) and global resources reservation (e.g., cloud storage and computing resources). To illustrate the proposed approach, we have chosen three inter-dependent services with varying characteristics which are quite typical in today's use of cloud services in private, community and business contexts: *(1)* Social Network, *(2)* Cloud Storage and *(3)* Face Recognition, along with their dependency web services E-Mail and Image Processing.

The service dependencies results from the informal requirements of providers to link the creation of a service usage account to the presence of existing service accounts. For example, it is common to require a valid e-mail address to sign up for a social network, or a social network account to sign up for the image processing web service. Furthermore, some web applications are using personal cloud storage to persist application specific data which forces the user to specify a cloud storage account when signing up to one of those applications [5]. From a consumer perspective, cloud services follow a mediated lifecycle: Services are selected, configured and used through central management facilities. Recent crashes and data leaks of the cloud computing infrastructure of some providers (e.g., Amazon's and Dropbox' cloud portfolio) have shown that today's cloud computing facilities are not necessarily reliable. Furthermore, privacy issues are causing huge discussions because most users avoid to give away personal data. To deal with those problems, it is often advised to federate or to aggregate them from several providers into service offering variations. In particular, optimal cloud storage with guaranteeable security and availability properties can be achieved with redundant data dispersion across multiple providers [6]. In this way, stored data remains available, even if some cloud services are temporarily unavailable. To ensure privacy, data can be scattered and dispersed across several providers in a way that none of them alone can reassemble the original information. Hence, the proposed autonomous participation scheme considers both lifecycle-phase-bound service dependencies and service offering variations such as bundles and compositions.

The cloud service dependencies and service offering variants are shown in Fig. 1. Further invocation dependencies which may exist during the service use within compositions are not subject to our approach; existing comparisons of these are well-known [7], [8].

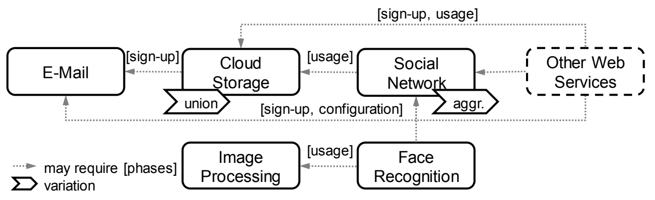

Figure 1. Dependencies between cloud services with and without typical offering variants

### B. Related Work

Research agendas towards a closer integration of multi-agent systems (MAS), service-oriented computing approaches and architectures (SOC/SOA) and autonomous/autonomic computing techniques have been published widely in the literature of their respective communities [9], [10]. Among the expected benefits are inter-agent communication guarded by service level agreements (SLAs), enhancement of agent knowledge and capabilities through services, and agent code reuse. New challenges arising from this combination include service procurement with uncertainty [11]. There are subtle differences between automation, increased autonomic self-management and full autonomy which we acknowledge in our work [2].

Early responses to the question of how to achieve autonomous service participation favoured a protocol-level connection approach between web services and software agents [12]. However, they did not account for deterministic service selection based on non-functional requirements and they omitted a discussion of how to integrate the connection into devices with a varying degree of autonomous operation. More recent approaches such as MySIM [13] introduce middleware-mediated spontaneous service selection via QoS specifications in pervasive applications. MySIM assumes users performing the service integration in a one-shot process and has not yet been evaluated in a context of binding existing cloud services to CPS. Another approach introduces local registries to avoid the costly (both in the resource and power consumption and in the financial sense) discovery of services [14]. However, it assumes interactive Internet devices controlled by humans in many decisions.

## III. Motivating Example

As a motivating example for autonomous participation of robots in cloud services and social networks, imagine the following scenario (cf. Fig. 2): At a fair or exhibition stand, a humanoid robot is responsible for taking pictures of humans attending the exhibition and passing by the stand. The robot is able to store the pictures using dynamically selected cloud storage with a suitable configuration including size and storage redundancy, as well as acting as a virtual reporter by selecting pictures, adding comments and posting them in a social network to document the exhibition day. Besides, visitors can use their own social network accounts to become friends of the robot. Afterwards, the robot will use their information to filter interesting pictures from its picture store (e.g., the robot is able to notify visitors, if it identifies colleagues or friends of them attending the exhibition as well). This scenario clearly summarises the requirements for an autonomous participation scheme: Dynamic service selection and variant handling, service dependency exertion and service integration into a feedback loop to find out if the chosen service combinations truly fulfil the requirements.

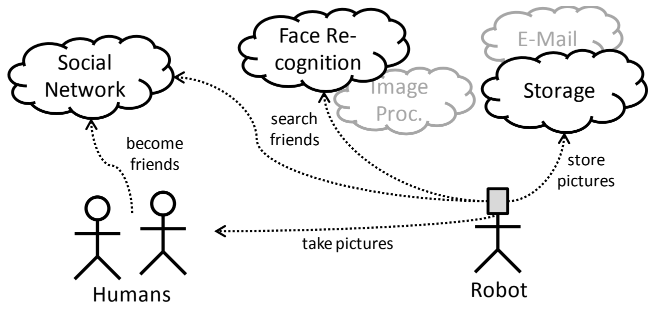

Figure 2. Example for autonomous participation of robots in social networks and other cloud services

## IV. Autonomous Participation

In order to autonomously participate in cloud services, several automated support activities are required along the cloud service consumption lifecycle. They encompass the sign-up preparation, the sign-up execution, and the continuous service usage, which is not covered in this paper. Software agents are leveraged to automatically execute each of the lifecycle steps autonomously and context-dependent [15].

- Sign-up preparation. This includes the selection of an identity and of requirements as bootstrap data followed by the service discovery and selection.
- Sign-up execution. E-Mail is selected or signed up first; other services like Social Network, Cloud Storage and Image Processing follow.
- Continuous usage (not considered in this article). This encompasses communication activities by E-Mail and in Social Networks, data storage, backups and sharing over Cloud Storage, as well as submission of relevant camera data to the Image Processing.

To harmonise the software agent deployment with service-oriented architectures dominant in cloud environments, we propose to store the agent implementations or links to them along with the service descriptions in a registry. At sign-up time, the agents are parametrised with an identity and with context information, which might be queried from a user explicitly or taken from a device context, about the desired service configuration. For cloud storage, an essential property is the capacity; for social networks, the privacy settings. Smart Application Grids (SMAGs) is an approach to build component based systems that can dynamically be adapted to the external environment using (object-)roles [16]. SMAGs put emphasis on dynamically changing collaborations between components within one and between several applications. In this way SMAGs are suitable candidates to implement and execute agents, especially when a cooperation between them is instrumental in achieving evolutionary adaptation of software and devices with long lifecycles to changing cloud environments. A feedback facility must be included for non-automatable sign-up steps (e.g., for solving CAPTCHAs [17]).

Apart from human users, both digital and physical cyber agents acting on behalf of or as substitutes of humans benefit from the autonomous integration of cloud services. These extended software agents represent the upper layer of the autonomous participation scheme and lift the degree of autonomy to the decision level. While humans typically start with a given identity (real or pseudonym) and conscious requirements, the cyber agents either need to be given this bootstrap data or given the facilities to generate it. With increasing capabilities of the agents, they can take over participation tasks from humans, for instance in the case of lost interest or death, as proposed by the Cyber-Individual model [18]. In addition to the system's sensors which trigger certain participation events, cyber agents can do so without a signal from outside, for instance in advance, by applying artificial intelligence, belief-desire-intent (BDI) patterns and needs-based triggers. Eventually, eliminating reasons to distinguish between humans and cyber agents will make cloud participants candidates for passing the Turing test [19]. In fact, even reverse Turing tests like CAPTCHAs against automation can be solved with properly coordinated automation, including automatic and human solvers at very low cost [17]. Fig. 3 visualises the service participation scheme AdAPtS including the relations between the autonomous participation engine, which hosts parametrised automating software agents, and the human and cyber participants which directly or through AdAPtS access services in the cloud.

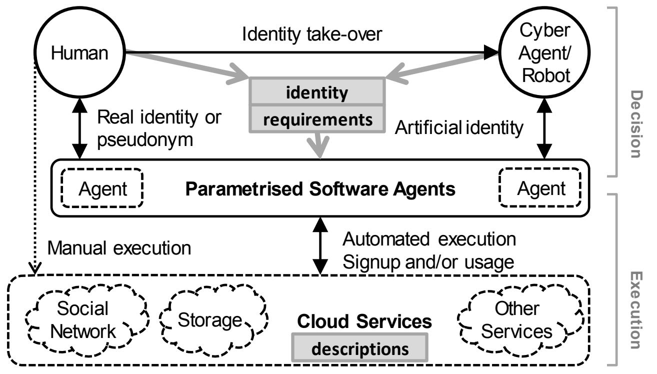

Figure 3. AdAPtS: Devices controlled by humans and cyber individuals participating in cloud services with a high degree of autonomy

## V. Realisation and Outlook

### A. Automated Service Integration

In order to demonstrate the usefulness of AdAPtS, we have first developed an automation framework for the sign-up to various semantically described cloud services [20]. All service property descriptions are represented as XaaS ontologies in the Web Services Modelling Language (WSML) and published into a semantic web service registry such as ConQo which is used for dynamic service discovery at runtime [21]. WSMO4IoS is an emerging public WSML ontology collection for both fictional and real XaaS with software, hardware and human implementation. We use its contents as the baseline for our prototype because it already allows us to differentiate early between suitable and unsuitable (e.g. expensive or mostly unavailable) services. The essential properties for automated service integration include the service name, either a fully qualified internationalised resource identifier (IRI) or unqualified, and endpoints to perform the service operations, starting with the sign-up. For services with a representation as human-readable web pages, the endpoints refer to those pages containing the forms to initiate the operations. An excerpt from an actual e-mail service description is contained in Listing 1. Its quantified non-functional properties like *Capacity* are matched against client requirements, whereas non-quantified ones like *Url* just carry information.

Listing 1. Semantic service description

```
instance Capacity memberOf { email#Capacity,
    qos#ServiceSpec }
        qos#value hasValue 1.5
        qos#unit hasValue qos#GB

instance GMXFreeMail memberOf { email#Email }
        hasUrl hasValue "pop3://pop.gmx.de"
        hasSignupForm hasValue "https://www.
            gmx.net/mail/freemail/
            registrierung/?0&mc=fm@hp@reiter.
            fm"
        hasCapacity hasValue Capacity
```

According to the registry model, each service entry can reference arbitrary description and integration artefacts. Typically, these are SLA templates, user interface bindings or functional interface descriptions (WSDL, WADL and similar formats). The automation framework uses this feature by referencing executable software agents for each phase of the service lifecycle. When a service operation (sign-up, usage etc.) is due, the agent description is parsed for dependencies whose agents are then executed first within the context of the framework before the agent for the relevant service itself is executed. Each agent produces a result (account data, return value etc.) which can be used by subsequent agents. The use of ConQo offers the benefit of being able to incrementally synchronise the local registry with a global one on demand whenever appropriate, e.g. in maintenance windows, to avoid costly runtime discovery processes.

### B. Robotic demonstration platform

In our presented scenario, the sign-up automation framework along with a local instance of the service registry and further software tools is hosted on a humanoid robot. We chose the Nao robotic platform, a popular standard platform for international competitions and research projects including autonomous behaviour and autonomic computing [22]. We have extended the Nao software tools with a custom installer for all of our software so that the following observations can be reproduced[1].

According to the scheme, the robot as CPS device needs an identity and a requirements specification before it can enact autonomous service participation. The Nao is hence given an initial identity, and it derives the need for service integration with all particular details through software and hardware sensors. For instance, its need for storage capacity is determined by measuring the occupancy state of the disk. Other non-functional properties such as price (for free) are still fixed but eventually be traded off for more capacity by linking the identity to a payment scheme which is reflected in the robot's identity. By matching the requirements against the services offered in the ConQo registry in a discovery and matchmaking procedure which includes a functional domain selection and unit normalisation, a ranked list of fitting services is returned. The re-use potential for the agents on just slightly varying services increases by applying SMAG, hence lowering the manual effort for autonomous continuous cloud service use.

Listing 2 is an example of an agent description for a specific cloud storage provider. Each agent executable is pre-scanned for a declarative list of requirements on the identity, for instance, the presence of a full name and a postal address. Hence, among all suitable agents for otherwise equivalent services, the one with the highest privacy index will be used. If on the other hand the identity is incomplete with regards to all available agents, the missing fields will be queried interactively or produced automatically by an incremental invocation of the CPS control's identity generator until a certain privacy threshold. The framework keeps track of all identity submissions. The data from such traces can be analysed to estimate the system's service activity and footprint.

Listing 2. Agent description

```
<serviceagents service="SafeSync" iri="http
    ://localhost:8080/Matchmaker/ontologies/
    CloudStorage/safesync.wsml#">
 <framework name="osst">
  <signup>
```

[1]Nao Cloud-Robotics installer: http://serviceplatform.org/cgi-bin/gitweb.cgi?p=smartoffice;a=tree;f=cloudrobotics

```
    <dependency domainiri="http://localhost
        :8080/Matchmaker/ontologies/Email.
        wsml#">
    <agent>http://localhost/agents/SafeSync.
        pm</agent>
  </signup>
 </framework>
</serviceagents>
```

## C. Autonomic Behaviour

The upper layer of the participation scheme has been realised with a needs-driven software brain for the robotic device following Maslow's theory of hierarchical needs ranging from fundamental to self-actualisation. First, in case the robot hasn't been given an identity yet, it will use a tool to generate and persist one which is indistinguishable from a human one. Then, all essential needs such as sufficient power supply are to be fulfilled, which partially relies on non-service physical actions, for instance moving to the closes power supply unit. Whenever this state is reached, certain sensoring and actuation tasks are performed within a control loop to ensure the system health and vividness. In idle periods of this state, the highest state is reached in which the robot could make up its own agenda based on the offers on a service marketplace. Our work doesn't cover this final state.

On the implementation level, the scheme has been realised with a high degree of re-use of existing software and data formats. Service descriptions have been taken from the public WSMO4IoS repository. The agents use the Mechanize and Tesseract-OCR frameworks to automate web page crawling and analysis tasks. The identity generation tool builds on top of the Barnum generator which is able to produce vCards for U.S. and German identities. Fig. 4 summarises the overall resulting architecture.

## D. Summary and Outlook

Through the research presented in this article, we have introduced a significant contribution to client-side service management. In evolving and unstable environments with various providers in the cloud, the solution minimises software reengineering efforts. Our methodology joins topics of high interest on the intersection of Autonomic Computing, CPS, and Cloud and Service Science research: lifecycle-phase-bound service dependencies, bootstrap data for the service selection in autonomous cyber agents and the interaction with the feedback loop through both client-specific sensors and CPS control agendas. The resulting autonomous participation scheme AdAPtS takes these concerns into account and enables a range of devices with a selectable degree of autonomy to access cloud services, such as self-healing cloud storage gateways and zero-configuration smart office applications. The AdAPtS components have been implemented and a video of the robot demonstrator has been made available[2]. Future work is scheduled to highlight the adaptive client-side integration concerns through the SMAG framework and long-term measurements of a continuously running CPS regarding the degree of configuration stability and self-determined recovery after service interruptions.

## Acknowledgements

This work has received funding under project numbers 080949277 and 080951806 by means of the European Regional Development Fund (ERDF), the European Social Fund (ESF) and the German Free State of Saxony.

## References

[1] M. Zeller, C. Prehofer, G. Weiss, D. Eilers, and R. Knorr, "Towards Self-Adaptation in Real-Time, Networked Systems: Efficient Solving of System Constraints for Automotive Embedded Systems," in *Proceedings of the 5th IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, October 2011, pp. 79–88, Ann Arbor, Michigan, USA.

[2] W. Truszkowski, L. Hallock, C. Rouff, J. Karlin, J. Rash, M. G. Hinchey, and R. Sterritt, *Autonomous and Autonomic Systems*. Springer-Verlag, 2009.

[3] Y. Brun, G. Marzo Serugendo, C. Gacek, H. Giese, H. Kienle, M. Litoiu, H. Müller, M. Pezzè, and M. Shaw, "Software engineering for self-adaptive systems," B. H. Cheng, R. Lemos, H. Giese, P. Inverardi, and J. Magee, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, ch. Engineering Self-Adaptive Systems through Feedback Loops, pp. 48–70.

[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[5] A. Narayanan, S. Barocas, V. Toubiana, H. Nissenbaum, and D. Boneh, "A Critical Look at Decentralized Personal Data Architectures," February 2012, Position paper, arXiv:1202.4503.

[6] J. Spillner, G. Bombach, S. Matthischke, J. Müller, R. Tzschichholz, and A. Schill, "Information Dispersion over Redundant Arrays of Optimal Cloud Storage for Desktop Users," in *4th International Conference on Utility and Cloud Computing (UCC)*, December 2011, pp. 1–8, Melbourne, Australia.

[7] R. Tolksdorf, "A Dependency Markup Language for Web Services," in *Web, Web-Services, and Database Systems. Proceedings of NODe 2002, Web- and Database-Related Workshops*, ser. Lecture Notes in Computer Science, October 2002, pp. 129–140.

[8] D. Retkowitz and S. Kulle, "Dependency Management in Smart Homes," in *Distributed Applications and Interoperable Systems, 9th IFIP WG 6.1 International Conference (DAIS)*, ser. LNCS, vol. 5523, June 2009, pp. 143–156, Lisbon, Portugal.

[2]Autonomous cloud service participation video: http://www.youtube.com/watch?v=NDlN0fG9Okk

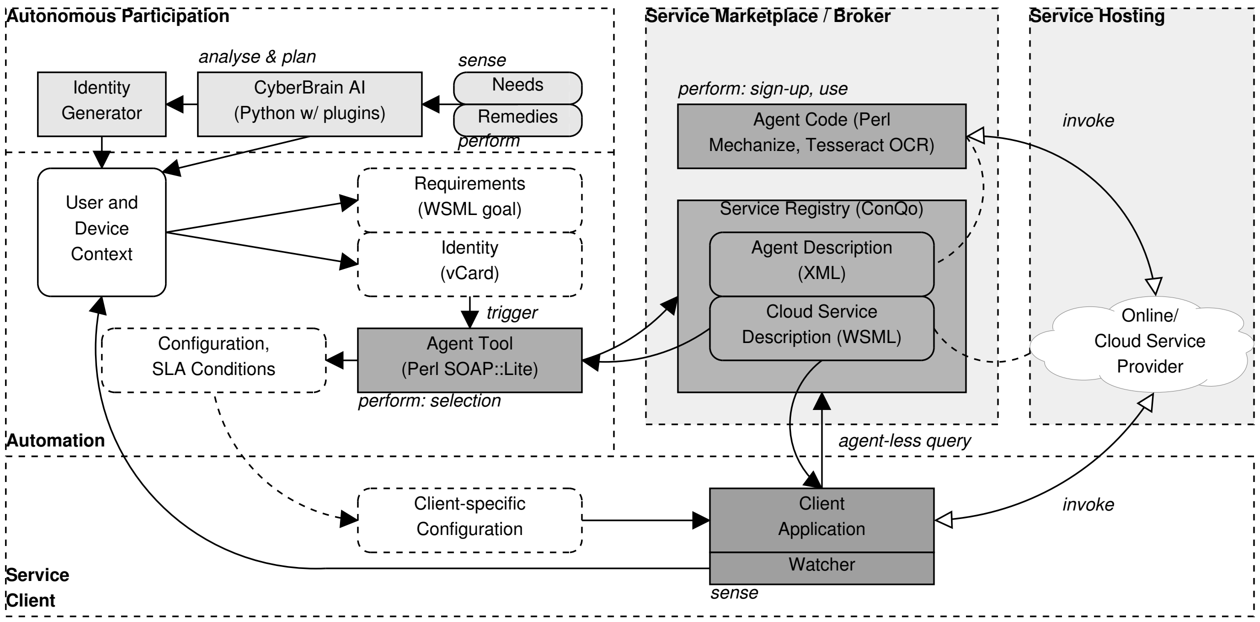

Figure 4. Automation and autonomous participation components added to a typical brokered service architecture

[9] F. M. T. Brazier, J. O. Kephart, H. V. D. Parunak, and M. N. Huhns, “Agents and Service-Oriented Computing for Autonomic Computing: A Research Agenda,” *IEEE Internet Computing*, vol. 13, no. 3, pp. 82–87, May 2009.

[10] D. Greenwood, M. Lyell, A. Mallya, and H. Suguri, “The IEEE FIPA Approach to Integrating Software Agents and Web Services,” in *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, May 2007, pp. 1412–1418, Honolulu, Hawai’i, USA.

[11] E. Gerding, S. Stein, K. Larson, A. Rogers, and N. R. Jennings, “Scalable Mechanism Design for the Procurement of Services with Uncertain Durations,” in *Proceedings of the 9th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, May 2010, pp. 649–656, Toronto, Canada.

[12] D. Greenwood and M. Calisti, “Engineering Web Service - Agent Integration,” in *IEEE International Conference on Systems, Man and Cybernetics (SMC)*, vol. 2, October 2004, pp. 1918–1925, The Hague, The Netherlands.

[13] N. Ibrahim, F. L. Mouël, and S. Frénot, “MySIM: a spontaneous service integration middleware for pervasive environments,” in *Proceedings of the ACM International Conference on Pervasive Services (ICPS)*, July 2009, pp. 1–10, London, United Kingdom.

[14] H. J. La and S. D. Kim, “A Service-Based Approach to Designing Cyber Physical Systems,” in *IEEE/ACIS 9th International Conference on Computer and Information Science (ICIS)*, August 2010, pp. 895–900, Kaminoyama, Yamagata, Japan.

[15] F. Zambonelli, N. R. Jennings, and M. Wooldridge, “Developing multiagent systems: the Gaia Methodology,” *ACM Transactions on Software Engineering and Methodology*, vol. 12, no. 3, pp. 317–370, July 2003.

[16] C. Piechnick, S. Richly, S. Götz, C. Wilke, and U. A. mann, “Using Role-Based Composition to Support Unanticipated, Dynamic Adaptation - Smart Application Grids,” in *Proceedings of ADAPTIVE 2012, The Fourth International Conference on Adaptive and Self-adaptive Systems and Applications (To appear)*, July 2012, Nice, France.

[17] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage, “Re: CAPTCHAs – Understanding CAPTCHA-Solving Services in an Economic Context,” in *19th USENIX Security Symposium*, August 2010, pp. 435–462, Washington, DC, USA.

[18] J. Ma, J. Wen, R. Huang, and B. Huang, “Cyber-Individual Meets Brain Informatics,” *IEEE Intelligent Systems*, vol. 26, no. 5, pp. 30–37, September/October 2011.

[19] A. M. Turing, “Computing Machinery and Intelligence,” *Mind*, vol. 59, no. 236, pp. 433–460, October 1950.

[20] C. Master, “Online Service Signup Tool,” Software project, online: github.com/cloudmaster/osst, 2012.

[21] J. Spillner, A. Kümpel, S. Uhlig, I. Braun, and A. Schill, “An Integrated Provisioning Toolchain for the Internet of Services,” in *Proceedings of IADIS International Conference WWW/Internet*, November 2011, pp. 566–570, Rio de Janeiro, Brazil.

[22] T. Niemüller, A. Ferrein, G. Eckel, D. Pirro, P. Podbregar, T. Kellner, and C. R. und Gerald Steinbauer, “Providing Ground-Truth Data for the Nao Robot Platform,” in *RoboCup 2010: Robot Soccer World Cup XIV*, ser. Lecture Notes in Computer Science, vol. 6556, 2011, pp. 133–144.