Towards Social Crowd Environments Using Service-Oriented Architectures

Soziale Crowd Umgebungen mittels Service-orientierter Architekturen

Florian Skopik, Daniel Schall, Harald Psaier, Vienna University of Technology, Austria,
Martin Treiber, IKANGAI Solutions e. U., Vienna, Austria,
Schahram Dustdar, Vienna University of Technology, Austria

Summary Crowdsourcing has emerged as an important paradigm of human problem solving techniques on the Web. More often than noticed, organizations outsource tasks to humans which cannot be processed by software. In this work we demonstrate the application of service-oriented architectures (SOA) for enterprise crowdsourcing. Crowdsourcing applications typically utilize the capabilities of people in open and dynamic Web-based systems. We extend this concept and introduce knowledge crowds where crowd members collaborate in context of joint tasks. Interactions in such environments are performed using software services. Such highly dynamic socio-technical environments, however, demand for flexible interaction models due to varying interaction styles of people and services. Our main contribution centers around the convergence of process artifacts and emerging social structures. Here, we present the implementation of a real world example, a human assisted image processing service that is provided by a knowledge crowd. We discuss the foundational building blocks for realizing the design, execution, and adaptation of service-oriented crowdsourcing applications. ►►► **Zusammenfassung** Crowdsourcing hat sich als wichtiges Paradigma etabliert, um menschliche Fähigkeiten zur Problemlösung über das Web anzubieten. Häufiger als allgemein bekannt lagern Organisationen Aufgaben auf externe Personen aus, insbesondere jene, die schwer von Software gelöst werden können. In dieser Arbeit demonstrieren wir die Anwendung von Service-orientierten Architekturen (SOA) für Crowdsourcing-Unternehmen. Crowdsourcing-Anwendungen nutzen typischerweise die Fähigkeiten von Menschen mittels offener und dynamischer Web-basierter Systeme. Wir erweitern dieses Konzept um die Einführung von Wissensbasierten Crowds, also Systemen in denen Mitglieder im Kontext von gemeinsamen Aufgaben kollaborieren. Interaktionen in solchen Umgebungen werden mittels Software-Services realisiert. Solch hoch-dynamische sozio-technische Umgebungen verlangen, insbesondere aufgrund wechselnder Interaktionsmöglichkeiten und -mustern, jedoch nach flexiblen Interaktionsmodellen. Unser Beitrag konzentriert sich auf die Konvergenz von Prozessartefakten und entstehenden sozialen Strukturen. Wir präsentieren die Umsetzung eines realen Anwendungsfalles, konkret ein Service zur Personen-gestützten Bildverarbeitung, welches von einer Wissensbasierten Crowd angeboten wird. Wir diskutieren die grundlegenden Konzepte für die Realisierung, die Laufzeit und periodische Anpassung einer solchen Service-orientierten Crowdsourcing-Anwendung.

Keywords C.2.4 [Computer Systems Organization: Computer-Communication Networks: Distributed Systems] Distributed applications; H.3.4 [Information Systems: Information Storage and Retrieval: Systems and Software] Information networks; H.3.5 [Information Systems: Information Storage and Retrieval: Online Information Services] Web-based services; H.4 [Information Systems: Information Systems Applications]; Crowdsourcing, Service-oriented Collaboration Systems, Human-assisted Computation ►►► **Schlagwörter** Aufgabenauslagerung, Service-orientierte Kollaborationssysteme, Personen-gestützte Bearbeitung

1 Introduction

Service-oriented architectures (SOA) represent a well-established paradigm to implement distributed enterprise computing systems [4]. SOA-based systems are typically realized based on Web services standards such as the Web Services Description Language (WSDL) and standardized message formats such as SOAP. These technologies enable the automation of three essential steps: (i) *publishing* of services and related artifacts, (ii) a *discovery* procedure to find services matching the requirements of a search query, and (iii) *interactions* to compose and aggregate the results of distributed software services. While the service-oriented computing approach has been designed for the purpose of automating interactions between software services only, we advocate a novel approach that utilizes *human capabilities* in a service-oriented manner. We consider mixed service-oriented systems based on the capabilities of Software-Based Services (SBS), which are fully automated services and Human-Provided Services (HPS) [11] for interfacing with people in a flexible service-oriented manner. In contrast to other approaches in workflow-based systems (e. g., WS-HumanTask [5]), we consider *open systems* wherein services can be added at any point in time. As an example, people may provide services by using tools similar to those found in Web 2.0 based collaboration environments such as Mashup editors to create lightweight compositions.

This view on SOA considering human and software-based services is motivated by applications found in emerging *crowdsourcing* environments. Task-based platforms for human computation and crowdsourcing [6; 14], including Amazon Mechanical Turk[1], CrowdFlower[2], Google's Smartsheet[3], or Yahoo's Predictalot[4] enable access to the manpower of thousands of people on demand by creating human-tasks that are processed by the crowd. Human-tasks include activities such as designing and testing products, voting for best results, or organizing information. The limitations of these platforms are that they do not provide (a) SOA-based principles to interface with people in a service-oriented manner and (b) complex interaction models between people to work on joint task assignments. For example, people in the crowd may need to delegate tasks to some other member of the crowd. While such interaction models are well established in traditional workflow systems, interaction models within the crowd have not been sufficiently addressed by existing platforms. Since crowdsourcing follows the open world assumption by letting people register their skills and capabilities in a service-oriented manner, assumptions made in traditional SOA need to be revisited.

Crowdsourcing potentially offers a powerful model for the discovery and inclusion of external experts in business processes because the allocation of the actual workforce is done *on-demand*. The challenge is that no assumption can be made about skills and incentives of the people contributing to outsourced tasks and service requests. Even if crowdsourcing seems convenient and attracts enterprises with scalable workforce and multilateral expertise the challenges of crowdsourcing are a direct implication of human's ad-hoc, unpredictable behavior and variety of interaction patterns.

In this work, we discuss an approach to *knowledge crowds* whose interactions evolve based on skills, performed tasks, and social preferences. We extend SOA concepts to enable dynamic interactions between crowd members. Compared to related approaches in crowdsourcing, our approach follows a new socially-based computing paradigm. Therefore, we introduce the concept of *crowdcomputing*. Following the open world assumption, humans actively shape the availability of services (HPSs). Social implications caused by human participation pose additional challenges to designing large-scale mixed service-oriented systems. Here we deal with the following contributions:

Crowdcomputing Scenario. We illustrate the application of crowdcomputing by discussing a real-world application scenario. In the given scenario, people perform tasks related to *human assisted image processing*.

Crowd Processes. We introduce tools to design processes in a top-down manner that are fully executed in the crowd. These processes are composed accounting for emerging social interaction patterns.

Social Interaction Patterns in Crowds. Since the proposed environment enables interactions between crowd members, e. g., requesting support or delegating tasks, we introduce an interaction mining approach in order to recognize common (successful) patterns (reflecting best practices) to be re-used in future collaborations.

Social Discovery and Compositions. Investigating dependencies between processed artifacts is the basis for finding reliable compositions of actors that can perform corresponding tasks. The concept of social trust supports the discovery of suitable crowd members.

2 Crowdsourcing Scenario

The presented example scenario illustrates a process that comprises tasks being outsourced to the crowd. In our example, the company *ikangai solutions*[5] provides a service which transforms vector graphics or bitmap images to Quartz 2D Objective-C code for use on the Apple iPhone[6]. The different available devices with varying form factors (screen size and resolution) require *manual*

[1] http://www.mturk.com/

[2] http://crowdflower.com/

[3] http://www.smartsheet.com/

[4] http://pulse.yahoo.com/y/apps/vU1ZXa5g/

[5] http://www.ikangai.com

[6] http://developer.apple.com/iphone/

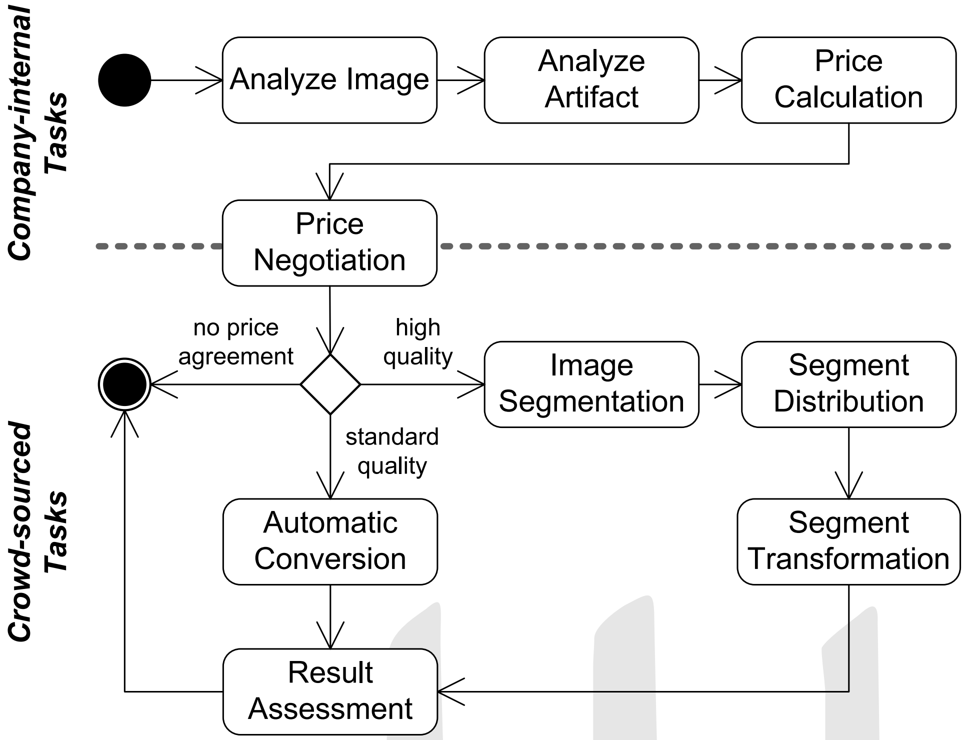


Figure 1 Human assisted image processing (HAIP).

adaptation and tailoring of graphics to the properties of a particular device.

We show a simplified version of the actual process in Fig. 1. For simplicity, we only illustrate the essential steps of the process flow without discussing detailed price negotiation procedures. The first step is *Analyze Image*. Here, a software tool checks whether the image (for example, a received bitmap) needs to be converted to a vector image. After that a vector representation [8] is used to determine the complexity of the image considering the number of vertices, edges and fillings (*Analyze Artifact*). The costs (*Price Calculation*) are estimated based on the desired quality. The *Price Negotiation* step has various alternatives. If an agreement with the customer about the price can be reached, the image is either transformed automatically by a software tool (standard quality depicted by the step *Automatic Conversion*) or processed by the crowd by performing the steps *Image Segmentation*, *Segment Distribution* and *Segment Transformation*. In either case, the result is checked by an ikangai employee (*Result Assessment*). Alternatively, the process is being aborted.

The steps *Image Segmentation*, *Segment Distribution* and *Segment Transformation* (see Fig. 1) are activities performed in the crowd during the execution of the transformation process. The actual allocation of the people/services for the transformation activity depends on their *expertise*: experienced people receive more complex segments to transform into Objective-C code, while less experienced crowd members receive considerable less complex segments to transform into Objective-C code. The allocation is done by a supervisor (also member of the crowd) who distributes the segments accordingly. Furthermore, since the process is also used to transform several bitmaps at once (e. g., application screen backgrounds), overlaps in the segments can be observed. Consequently, there is the need for *coordination and collaborations in the crowd*, i. e., direct interactions. For example, a segment containing a triangle that is used at different positions on different screens needs all crowd members to coordinate their activities to produce a composition of their work.

The actual degree of human involvement depends on the desired quality by the customers and the available resources, i. e., humans in the crowd. If high quality work is required, each transformation step is assisted by a human who ensures the required quality. For example, automated transformations typically do not optimize the lines of Quartz 2D code. Code that draws a rectangle with four lines and a fill can be replaced by a human with one line of code. And finally, once the result of the image conversion is received from the crowd, a company employee assesses the final result (*Result Assessment*) and checks the resulted Quartz 2D code again.

3 Social Crowd Management

We introduce a conceptual layered framework that motivates the need for various building blocks in order to address most common challenges of managing knowledge crowds.

The proposed approach consists of four different layers and distinguishes between design and emergence perspec-

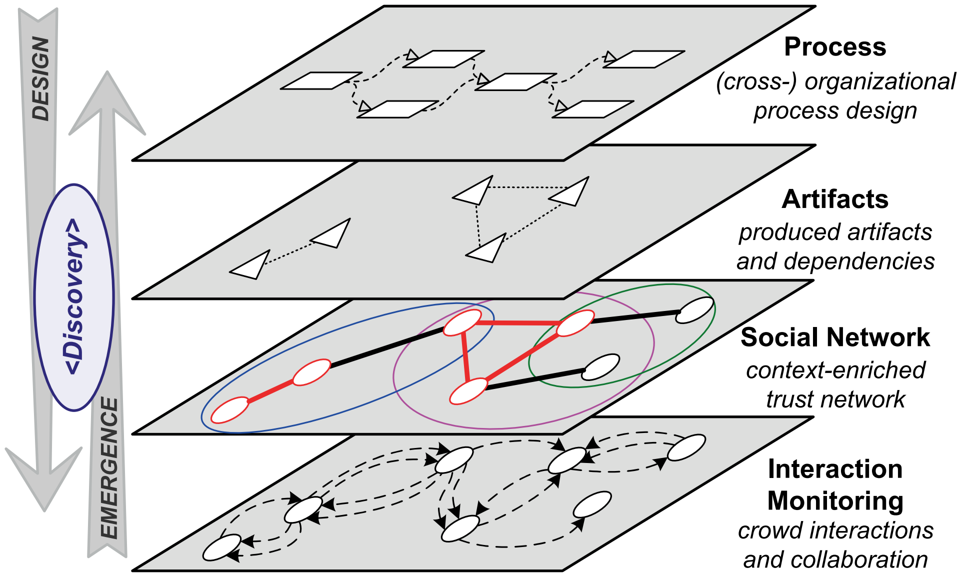

DESIGN
EMERGENCE
<Discovery>
Process
(cross-) organizational process design
Artifacts
produced artifacts and dependencies
Social Network
context-enriched trust network
Interaction Monitoring
crowd interactions and collaboration

Figure 2 Layered crowd management framework.

tive. Figure 2 shows the layered model. From a *design perspective*, one creates a process potentially spanning numerous organizations and individuals. In one or several tasks of this process, particular artifacts are created and processed respectively, whereas these artifacts may have various dependencies; for instance, consider the example scenario of this work where crowd members work jointly on the same image artifacts. From an *emergence perspective*, SOAP based interactions are observed and logged in order to enable the automatic inference of social relations; e. g., social trust [12]. Hence, based on interactions, such as collaborating in joint tasks, social networks form.

Various works have shown [7; 13] that there are strong dependencies between the coupling of team members and the artifacts they produce. In detail, the tighter people work together the stronger their corresponding artifacts are coupled; e. g., distributed software modules implemented by a distributed team. As an implication, we argue that tighter coupled artifacts (e. g., overlapping image segments) require stronger coupled team members *discovered* from the social trust network in order to meet a certain level of quality. In the introduced use case, the dependencies of artifacts that are created in context of crowdsourced tasks reflect the desired structure of social connections between people that are assigned to process these tasks. Notice, discovery mechanisms can also be applied in a bottom-up manner. By investigating emerging social structures and subcommunity relations, one can design processes and task flows that are optimized to be executed by particular crowd member compositions.

4 Knowledge Crowds

In this section, we discuss the flexible involvement of humans in service-oriented systems. We explain how humans add flexibility to existing processes and discuss how humans and crowd knowledge can be introduced to service-oriented systems. Second, the discovery of human capabilities needs to be supported using widely accepted formats such as enhanced Friend-Of-A-Friend (FOAF)[7] profiles. Notice, the focus of our approach is not to introduce or propose entirely new message formats or standards. Instead, we focus on the application and extension of well-established standards (e. g., WSDL for describing service interfaces and SOAP-based messages for service-oriented interactions) in crowdsourcing scenarios.

4.1 Knowledge in Crowds

As discussed in the scenario, humans and software services are composed for the provision of a bitmap to Quartz 2D transformation service. Activities like this require expertise in different areas, like knowing Quartz 2D commands in the first place or being able to optimize automated generated code. This kind of knowledge is implicitly available in crowds, however the formalization of such knowledge for the exploitation is a challenging issue. In order to address this problem, we propose to use FOAF profiles that are flexibly annotated with different keywords defining expertise in certain domains. Applying this approach, we are able to express knowledge in crowds, which is used to integrate human expertise as services into processes. For example, in the working scenario, we require the crowd to have knowledge about Quartz 2D programming and vector image conversion represented by distinct keywords.

4.2 SOA-based Human Computation

Many service-oriented architectures comprise software services only. While SOA has proven to be successful

[7] http://xmlns.com/foaf/spec/

in B2B integration and composition scenarios, more and more collaboration and composition scenarios require interactions between human actors as well as software services. Current tools and platforms offer limited support for human interactions in SOA. We therefore introduced the HPS framework. The aim of the HPS framework [11] is to

- offer a service registry maintaining information related to human and software services
- enhance service-related information by describing human characteristics and capabilities
- define interaction patterns using Web services technology so that human actors can efficiently deal with interactions.

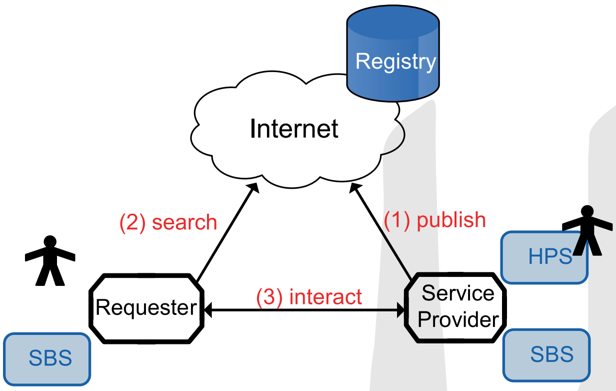


Figure 3 Enhancing SOA with human capabilities.

Human actors in crowdsourcing applications can therefore provide their capabilities and skills in a service-oriented manner. Following the SOA paradigm, three essential steps are performed:

1. *Publish.* Users have the ability to create HPSs and publish the services on the Web using a registry. Publishing a service is as simple as posting a blog entry on the Web. It is the association of the user's profile with an activity described as a service (WSDL). Interfaces provide the needed metadata support for the discovery of suitable HPSs.
2. *Search.* The service requester performs a keyword-based search (reflecting expertise areas) to find Human-Provided or Software-Based Services. Notice, also an HPS can act as a requester as services can be composed recursively (e. g., nested interactions through delegation mechanisms). Ranking is performed to find the most relevant HPS based on, for example, the expertise of the user providing the service. Expertise is determined automatically by the HPS framework through context-sensitive interaction mining techniques.
3. *Interact.* The framework supports automatic user interface generation using XML-Forms technology[8]. Thus, personalized interaction interfaces can be generated and rendered for different devices. The HPS framework can be used for interactions between humans and also for interactions between software services and HPSs.

4.3 Activity Artifact Dependencies

Artifacts that are created and processed by distinct crowd members, typically have some sorts of dependencies. Especially, if crowd-sourced tasks belong to a higher level process, relevant artifacts are often split, distributed, processed and then merged again. For instance, in the motivating use case, a bitmap figure may be split into several subfigures and processed by different crowd members at the same time; however, involved members need to coordinate their work, i. e., interact with each other. For that purpose, selecting people already having social relations and who know each others' working styles is beneficial instead of composing entirely new teams. In other words, the structure of artifacts and their dependencies should follow the social structure of selected crowd members in order to achieve highest possible reliability and success [7; 13].

4.4 Social Trust Network Management

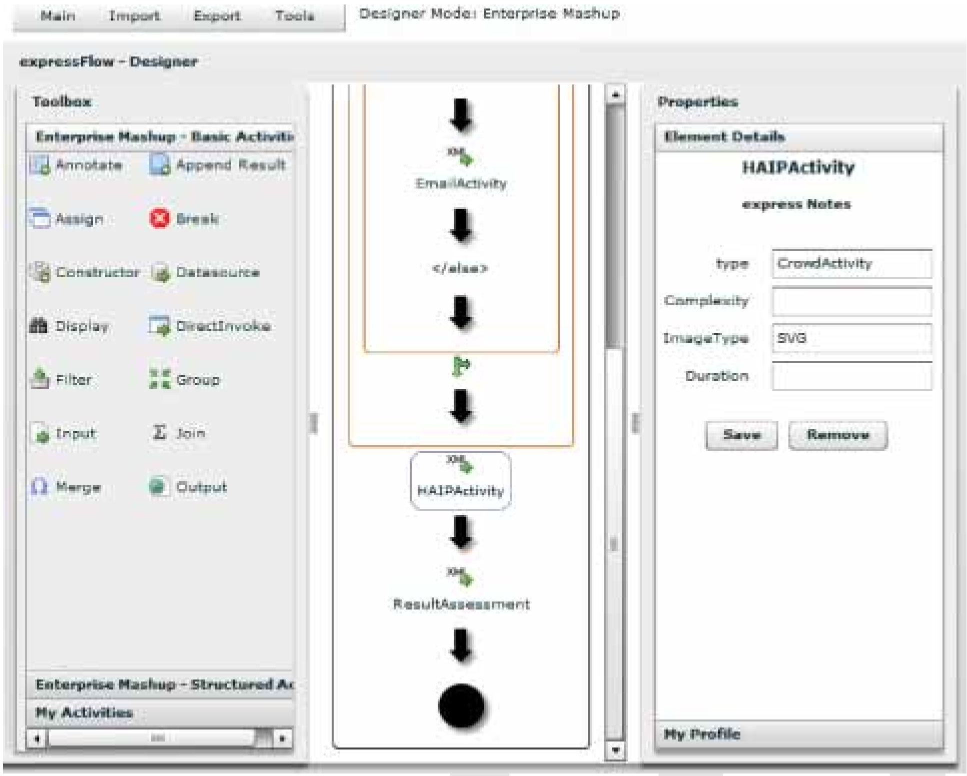
Nowadays, social networks are increasingly used to manage personal contacts and relations. Various models have been proposed, such as Friend-Of-A-Friend (FOAF), XHTML Friends Network[9] (XFN), and OpenSocial[10]. This paper deals with supporting formations and interactions in service-oriented crowd environments by accounting for the individuals' *social relations*, especially social trust. In contrast to a common security perspective on trust, the notion of social trust refers to the interpretation of previous collaboration behavior and the similarity of dynamically adapting interests (see [12] for details). Especially in collaborative environments, where users are exposed to higher risks than in common social network scenarios, and where business is at stake, considering social trust is essential to effectively guide human interactions.

The applied trust model manages context-dependent trust links between actors emerging from interactions that are captured and interpreted. A trust relation is always asymmetric, i. e., a directed edge from the *trustor* to the *trustee*. Analyzed *interactions* are any kind of communication, coordination or execution actions initiated by the trustor regarding the trustee. The *context* of interactions reflects the situation and reason for their occurrences, and is modeled as activities. Activities, as presented in [11], describe work-relevant context elements. In this model, relations are managed by the system. In contrast to manually declared relations, this approach allows to keep track

[8] http://www.w3.org/MarkUp/Forms/

[9] http://gmpg.org/xfn/

[10] http://code.google.com/apis/opensocial/



type: CrowdActivity
Complexity:
ImageType: SVG
Duration:

Figure 4 ExpressFlow online tool for process design.

of changing interaction behavior and interests even in highly dynamic networks.

5 Process Design

Typically one starts with designing a process from an organizational point of view, by defining and connecting single tasks. Later, process stakeholders decide which tasks are performed within a company and which ones are outsourced to a crowd environment.

5.1 ExpressFlow Process Design Tool

In the described environment, processes are designed using the Web-based ExpressFlow[11] editor. The editor, as shown in Fig. 4, has three panes: the left pane shows a toolbox with a set of basic activities. The middle pane shows the designed process flow consisting of different kinds of activities and structures to control the flow (branches, conditions, etc.). The right pane details, for example, specific types of activities.

In this example, the `HAIPActivity` that is part of the process, should be outsourced to the crowd. The three parameters of the activity are *Complexity*, *ImageType*, and *Duration*. These parameters are initialized during runtime through the execution context of the process.

5.2 Connecting the Crowd to Processes

ExpressFlow supports the generation of enterprise mashup code[12] (EMML) that can be deployed in an execution platform. The `HAIPActivity` is a container for *actions*. The action is, for example, a `<directinvoke>` as specified by EMML to invoke services (such as HPSs).

Listing 1 Service invoke example.

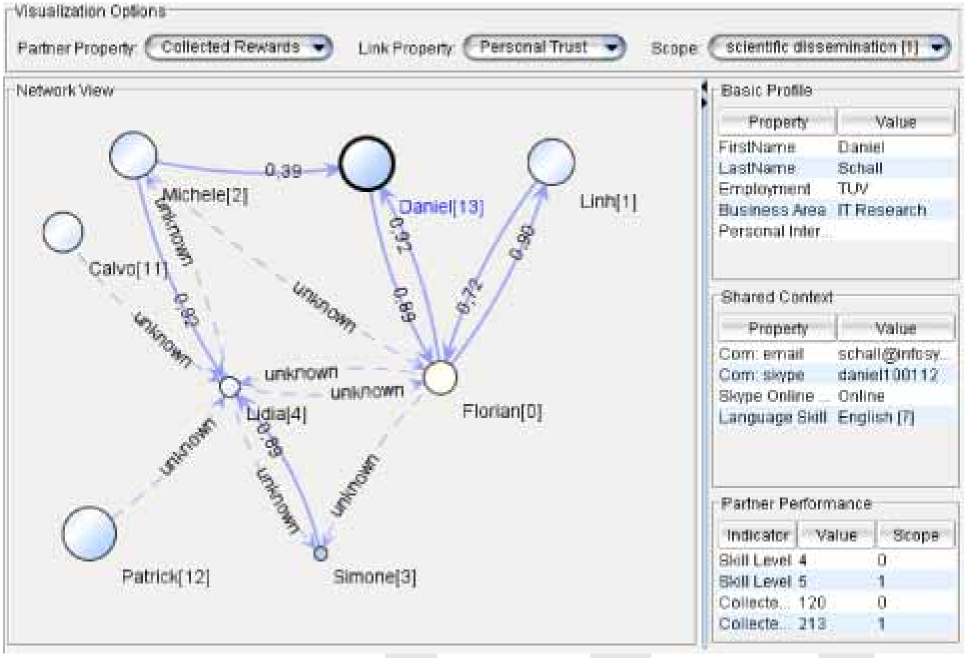
```
<directinvoke endpoint="$serviceURL" outputvariable="$actId"
  method="post" requestbody="$HAIPRequest" />
```

Before invoking the service, the corresponding SOAP envelop must be prepared by inserting the invocation parameters (based on the process execution context) as shown in Listing 2.

Listing 2 Sample SOAP envelope.

```
<constructor outputvariable="HAIPRequest">
<soap:Envelope xmlns:soap="http://.../soap/envelope/">
  <soap:Body>
    <ns2:addActivity xmlns:ns2="http://.../G2/generated/HAIP">
      <img>
        <mimeType>image/png</mimeType>
        <size>640x480</size>
        <uri>http://somerepository/path/to/file</uri>
      </img>
      <cmp>
        <type>code optimization</type>
        <degree>high</degree>
      </cmp>
      <dl>P1DT0H0M0S</dl>
    </ns2:addActivity>
  </soap:Body>
</soap:Envelope>
</constructor>
```

[11] http://expressflow.com
[12] http://www.openmashup.org/omadocs/v1.0/emml/



Visualization Options

Partner Property: Collected Rewards

Link Property: Personal Trust

Scope: scientific dissemination [1]

Network View

Basic Profile

Property	Value
FirstName	Daniel
LastName	Schall
Employment	TUV
Business Area	IT Research
Personal Inter...	

Shared Context

Property	Value
Com: email	schall@infosy...
Com: skype	daniel100112
Skype Online ...	Online
Language Skill	English [7]

Partner Performance

Indicator	Value	Scope
Skill Level	4	0
Skill Level	5	1
Collecte...	120	0
Collecte...	213	1

Figure 5 Discovery of emerging actor compositions.

6 Emerging Interactions

From a bottom-up perspective communication, coordination, and collaboration lead to the emergence of social relations. In particular, observable interactions enable the automatic management of such relations in a computational model.

6.1 Interactions and Trust Inference

Common approaches that use interaction mining to determine the strength of social relations, utilize an abstract notion of interaction. They collect very basic interaction data such as fundamental exchanges of e-mail messages, VoIP calls, and instant messages. Even when applying complex text analysis, it is hardly possible to determine the semantic meaning of these interactions, e. g., the reason for initiating a Skype call. However, we use *Web service enabled interactions*, using HPSs for delegating activities, performing periodic reports, and requesting help and support. That approach provides more information on the nature of collaboration and context of single interactions. Aggregating and interpreting interactions enable the calculation of interaction metrics, reflecting properties such as reliability, reciprocity, and responsiveness [12] within respective contextual constraints. These factors are the underlying basis for inferring evidence-based trust relations.

6.2 Discovery and Compositions

Crowd proxies are managing entities that receive sets of tasks (e. g., instances of `HAIPActivity`) from organizations to be outsourced to the crowd. Depending on their dependencies and the dependencies of created and processed artifacts respectively, that proxies discover suitable teams to delegate these tasks to. Crowd proxies may be represented by human operators or even implemented in software. Here, we focus on human roles.

The *Monitoring Dashboard* in Fig. 5 visualizes emerged social relations from previous interactions. Single crowd members are represented by nodes and directed edges connect them. These links are annotated with calculated interaction metrics and interpreted personal trust. As shown in Fig. 5, the crowd proxy can select visualized metrics from a toolbar. S/he is further able to resize, expand or collapse the network view, and thus, browse dynamically through the social network. Finally, a scope determines the context of previous collaborations, and thus, reflects expertise areas. Using the Monitoring Dashboard and its rich set of features, the crowd proxy is able to properly match artifact dependencies of received tasks to a particular social compositions of crowd members.

7 Related Work

Human provided services [11] close the gap between SBS and humans desiring to provide their skills and expertise in a service-oriented manner. In business environments (typically *closed* systems), human-based process activities (see BPEL4People [2]) and human tasks [5] can be modeled in a standardized manner. These standards, however, demand for a precise definition of roles and interaction models between humans and services. The application of such models is therefore limited in crowdsourcing scenarios since interaction flows in open and dynamic environments typically emerge at runtime. In Web-based systems, users share their expertise [15] or offer their expertise by helping other users in forums or answer communities [1]. Our approach is based on well-established standards such as WSDL for defining interfaces for HPS and FOAF-based social network profiles.

The availability of rich and plentiful data on human interactions in *social networks* has closed an important loop [9], allowing one to model social phenomena and to use these models in the design of new computing applications such as crowdsourcing techniques [6]. Semantic Web service communities as introduced by [10] foster the creation of structured communities with predefined community interfaces and functionality. However, ontology structures are not well suited for crowds, because crowd structures emerge bottom up and are difficult to capture with regard to functionality and interactions between crowd members. Also, value networks [3] are of interest when business aspects are investigated in crowd settings, i. e., the value that can be generated by such networks based on crowd capabilities and knowledge.

8 Conclusion and Future Work

In this paper, we proposed a novel approach for integrating human capabilities in crowd process flows. We focused on crowdsourcing applications by discussing a real world example. In the given scenario, people perform tasks related to human assisted image processing that clearly demand for the combination of human and software-based services. Our approach is based on both top-down process modeling and emerging interaction patterns in the crowd. We discussed fundamental problems and building blocks supported by service-oriented architectures. While traditional SOA focuses on the discovery of software-based services, we extend this notion with Human-Provided Services. In flexible interaction environments, one must consider social preferences and behavior patterns. Thus, interaction monitoring and context-aware discovery, even of compositions of actors, are essential to cope with inherent dynamics of large-scale social and collaborative environments.

Our future work considers more complex compositions of human actors and even software-based services in crowdsourcing scenarios. We are working on algorithms to discover suitable service compositions for given sets of artifacts, considering their dependencies and required actor skills. This will relieve crowd proxies from manually assigning tasks to particular crowd members. Furthermore, by automatically assigning tasks to crowd members, a more dynamic load balancing and failure compensation is enabled. Also, we are currently working on the integration of SLA (service level agreement) frameworks to model human quality attributes in the described environment.

References

[1] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne. Finding high-quality content in social media. In: *Proc. of the Int'l Conf. on Web Search and Web Data Mining (WSDM)*, pages 183–194. ACM, 2008.

[2] A. Agrawal et al. *WS-BPEL extension for people (BPEL4People), version 1.0.*, 2007.

[3] V. Allee. Reconfiguring the value network. In: *Journal of Business Strategy*, 21(4), Aug. 2000.

[4] G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web Services – Concepts, Architectures and Applications*. Springer, October 2003.

[5] M. Amend et al. *Web services human task (WS-HumanTask), version 1.0.*, 2007.

[6] D. Brabham. Crowdsourcing as a model for problem solving: an introduction and cases. In: *Convergence: The International Journal of Research into New Media Technologies*, 14(1):75, 2008.

[7] C. R. B. de Souza, S. Quirk, E. Trainer, and D. F. Redmiles. Supporting collaborative software development through the visualization of socio-technical dependencies. In: *Proc. of the 2007 Int'l ACM Conf. on Supporting Group Work (GROUP)*, pages 147–156. ACM, 2007.

[8] L. Eikvil, K. Aas, and H. Koren. Tools for interactive map conversion and vectorization. In: *Proc. of Int'l Conf. on Document Analysis and Recognition (ICDAR)*, pages 927–930, 1995.

[9] J. Kleinberg. The convergence of social and technological networks. In: *Communications of the ACM*, 51(11):66–72, 2008.

[10] B. Medjahed and A. Bouguettaya. A dynamic foundational architecture for semantic web services. In: *Distributed and Parallel Databases*, 17(2):179–206, 2005.

[11] D. Schall, H.-L. Truong, and S. Dustdar. Unifying human and software services in web-scale collaborations. In: *IEEE Internet Computing*, 12(3):62–68, 2008.

[12] F. Skopik, D. Schall, and S. Dustdar. Modeling and mining of dynamic trust in complex service-oriented systems. In: *Information Systems*, 35(7):735–757, Nov. 2010.

[13] E. Trainer, S. Quirk, C. R. B. de Souza, and D. F. Redmiles. Bridging the gap between technical and social dependencies with ARIADNE. In: *Proc. of the 2005 OOPSLA Workshop on Eclipse Technology Exchange (ETX)*, pages 26–30. ACM, 2005.

[14] M. Vukovic. Crowdsourcing for enterprises. In: *Proc. of the 2009 Congress on Services*, pages 686–692. IEEE, 2009.

[15] J. Yang, L. Adamic, and M. Ackerman. Competing to share expertise: the taskcn knowledge sharing community. In: *Proc. of the Int'l AAAI Conf. on Weblogs and Social Media (ICWSM)*, 2008.

Received: March 17, 2011

The authors from left: F. Skopik, H. Psaier, D. Schall, M. Treiber, S. Dustdar

Florian Skopik is a postdoctoral researcher working in the fields of reputation models and social trust management for flexible Web-based collaboration systems.

Address: Vienna University of Technology, Argentinierstrasse 8/184-1, 1040 Vienna, Austria, Tel.: +43-1-58801-58418, Fax: +43-1-58801-18491, e-mail: skopik@infosys.tuwien.ac.at

Daniel Schall is a postdoctoral researcher interested in techniques to program and control dynamic compositions in systems that are governed by complex interactions between human-provided and software-based services.

Address: Vienna University of Technology, Argentinierstrasse 8/184-1, 1040 Vienna, Austria, Tel.: +43-1-58801-18453, Fax: +43-1-58801-18491, e-mail: schall@infosys.tuwien.ac.at

Harald Psaier is a PhD student working on service-oriented systems using self-* concepts, including self-healing and self-adaptation principles.

Address: Vienna University of Technology, Argentinierstrasse 8/184-1, 1040 Vienna, Austria, Tel.: +43-1-58801-58418, Fax: +43-1-58801-18491, e-mail: psaier@infosys.tuwien.ac.at

Martin Treiber is the technical mastermind of IKANGAI Solutions, and specialized in the development of iPhone and iPod touch applications.

Address: IKANGAI Solutions e. U., Treustrasse 59/5/20, 1200 Vienna, Austria, Tel.: +43-1-8903172, Fax: +43-1-8903172,
e-mail: office@ikangai.com

Schahram Dustdar is full professor of computer science with a focus on Internet technologies heading the Distributed Systems Group.

Address: Vienna University of Technology, Argentinierstrasse 8/184-1, 1040 Vienna, Austria, Tel.: +43-1-58801-18414, Fax: +43-1-58801-18491, e-mail: dustdar@infosys.tuwien.ac.at