Supporting Multilevel Incentive Mechanisms in Crowdsourcing Systems: an Artifact-centric View

Ognjen Scekic, Hong-Linh Truong, and Schahram Dustdar

Abstract Crowdsourcing systems of the future (e.g., Social Compute Units — SCUs, collective adaptive systems) need to support complex collaborative processes, such as software development. This presupposes deploying ad-hoc assembled teams of human and machine services that actively collaborate and communicate among each other, exchanging different artifacts and jointly processing them. Major challenges in such environments (e.g., team formation, adaptability, runtime management of data-flow and collaboration patterns) can be somewhat alleviated by delegating the responsibility and the know-how needed for these duties to the participating crowd members, while indirectly controlling and stimulating them through appropriate incentive mechanisms. Existing process-centric collaboration modeling approaches (e.g., workflows) are incapable of encoding such incentive mechanisms. Therefore, in this paper we analyze different interaction aspects that incentive mechanisms cover and formulate them as requirements for future systems to support. We then propose an artifact-centric approach for modeling incentives in rich crowdsourcing environments that meets these requirements.

Ognjen Scekic
Distributed Systems Group, Vienna University of Technology, Austria
e-mail: oscekic@dsg.tuwien.ac.at

Hong-Linh Truong
Distributed Systems Group, Vienna University of Technology, Austria
e-mail: truong@dsg.tuwien.ac.at

Schahram Dustdar
Distributed Systems Group, Vienna University of Technology, Austria
e-mail: dustdar@dsg.tuwien.ac.at

1 Introduction

Many previous works on crowdsourcing seem to assume that *crowd* is an unlimited pool of adequate human workforce and typically focus on problems such as: locating the most appropriate candidates for performing the tasks, or comparing different payment schemes. However, while the assumption of the practically unlimited crowd may hold true in case of simple, independent tasks, the practice shows that the current crowdsourcing models (both from the technical and the from business perspective) fail to attract and retain workers capable of performing complex/modular, interdependent tasks[23, 27].

One of the reasons is that in the case of (highly-)skilled individuals, the crowd is, in fact, a quite limited resource pool that an ever increasing number of crowdsourcing efforts are trying to tap into. This means that the individuals need to be motivated through diverse, elaborate incentive and rewarding strategies to join a particular crowdsourcing effort and to provide their professional services at an expected level.

The other reason is that the existing crowdsourcing platforms do not offer flexible, human-like collaboration platforms to the crowd workers. Rather, the tasks are either assigned to the human workers by the system executing the workflow, or the humans bid for tasks on micro-task platforms. In both cases, the managing system dictates the orchestration, treating the crowd workers as machine computing elements, which are requested to respect the prescribed orchestration and various functional and quality constraints without being able to influence them[1].

This situation contradicts the very reason why humans are included into computations in the first place — to do better what computers are not good in doing — i.e., to bring in creativity, flexibility in unforeseen situations, but most importantly, ability to quickly perform complex tasks by establishing ad-hoc collaborations and adapting them when needed.

1.1 Motivation

To make humans first-class citizens, workers must be given more influence on selecting their collaboration partners, coordination patterns and communication channels. Hard constraints and worker commitment protocols should be loosened to make the systems more attractive for human workers. The price to pay for this is a degree of outcome uncertainty that must be reckoned with. We can either embrace it as an inherent property of these *socio-technical/crowdsourcing systems* (like we do in most everyday life situations) or try to blindly follow the conventional paradigms and seek to detect and/or correct those uncertainties. Embracing uncertainty, however, does not mean promoting it, but rather implies usage of different passive measures for reducing it to an acceptable level. This can be achieved through *incentive mechanisms* motivating workers to self-organize and self-correct. To this end, in

[1] See [5] for an overview of task distribution and coordination models

this paper we investigate the necessary requirements for defining and enacting such incentive mechanisms in the novel types of crowdsourcing systems [12].

To the best of our knowledge, incentives in crowdsourcing have so far been only considered at the granularity level (scope) of a business process (Sec. 4.2). As a business process typically contains a flow of different activities on multiple artifacts, workers can exhibit different behaviors depending on which activity they perform, on which artifact and with which co-workers. This means that the existing incentive models are successfully applicable only in a limited number of cases, where business processes are simple and dominated by a single activity. This is exactly the case with today's commercial crowdsourcing platforms that incentivize business processes that typically require a single worker to process and return an artifact (e.g., describe a bug, submit a design, tag a photo, translate a text). As these processes are simple, the incentives need only to focus on the core activity, and to promote wanted behaviors, like diligence and quality.

However, performing complex tasks, such as software development, with crowdsourced, ad-hoc teams involves many activities, workers and interactions that are not predictable in advance. Developers may come and go; their performance may vary; they may be using different tools to communicate, coordinate and produce code, tests and documentation; they may be used to different development methodologies and team organizations. It is not realistic to expect that a team formed out of such diverse individuals will adhere to a prescribed execution plan. In fact, designing a work process with so many unknowns will probably result in an inefficient workflow at runtime [3]. And without a valid workflow, it is impossible to design appropriate incentive mechanisms either.

Similar problems have previously been investigated by large traditional companies trying to impose uniform work processes across geographically distributed workforce. They discovered that different internal teams would agree more easily on a common set of artifacts to use in interactions rather than on the common activity flow [16]. This resulted in the birth of *artifact-centric workflows*[10]. The principal idea is to focus on data (artifacts), rather than on processes, and to leave the actors more freedom to self-organize, while controlling them indirectly through artifacts augmented with a formal lifecycle model. (see Sec. 4.1 for more information).

We believe that, if extended with incentive mechanisms, the artifact-centric approach can be successfully used to describe and guide complex crowdsourcing processes. Augmenting artifact models with incentive mechanisms creates entities that self-motivate people to process and control them. By attracting workers to work at them, the artifacts push their way through the lifecycle. In addition, if we encode incentive application rules at the artifact-granularity level, we can express much finer conditions. This opens up an array of new possibilities for motivating humans to work in crowdsourcing platforms.

1.2 Contributions and Article Structure

In this paper we propose applying artifact-centric approach to designing incentives for socio-techincal systems. We argue that this approach may be better suited than the traditional process-oriented approach, covering better the different possible aspects of human behavior and business needs in complex collabrative environments. We then identify the concrete requirements for designing such incentive systems.

The rest of the paper is structured as follows:
Section 2 introduces some fundamental notions that are used in the rest of the paper. The rest of Sec. 2 presents a motivating example, and uses it to highlight important aspects when designing incentive mechanism models for crowdsourced, artifact-centric workflows. In Sec. 3 we further analyze these aspects and identify important requirements for novel crowdsourcing systems supporting artifact-centric incentive mechanisms. In Sec. 4 we present a short review of related work on incentives in crowdsourcing and traditional artifact-centric workflows. Sec. 7 presents the summary and concludes the paper.

2 Artifact-centric Incentives

We begin by defining some important terms as used throughout this paper:

Definition 0.1 (Incentive). Any scheme employed by the system to stimulate (motivate) increased level of certain worker activities (e.g., productivity, speed, quality of work, number of participants) or to discourage certain activities (e.g., drop-out rate), before the actual execution of those activities.

Definition 0.2 (Reward). Any kind of recompense for worthy services rendered or retribution for wrongdoing exerted upon workers after the completion of the activity.

Definition 0.3 (Incentive Mechanism). A clearly delimited incentive rule targeting a specific dysfunctional behavior.

An incentive mechanism consists of the following three components [23]:

1) **Evaluation Method** — used to assess the quality of worker's performance from different aspects. Provides input for making a decision whether to apply a reward/sanction.
2) **Incentive logic** — represents the business logic behind the incentive mechanism used to interpret evaluation results and decide on application of rewarding actions.
3) **Rewarding Action** — represents the concrete measure taken against individuals or teams to influence one particular future behavior.

Definition 0.4 (Business Artifact). First-class entity of a business process encapsulating all the information necessary for its processing throughout the entire execution of the business process. The notion of artifact includes not only the 'raw'

data that is produced or processed during the business process, but also the metadata describing the lifecycle, relationships with other artifacts and context-dependent information.[2]

The artifacts are identified and described by domain experts. They can correspond to the actual (physical or digital) entities used by the participants in a business process (such as invoices, bills, source code files, commitment history), or be abstract entities that facilitate the process execution management.

Apart from the obvious purpose of capturing (intermediate) business process goals, each artifact is also supposed to capture the information for evaluating if and how well the goals have been achieved. For example, in addition to the description of the problem and associated fix code, a software bug report artifact may contain the history of actions taken, allowing to draw conclusions on the quality and speed of the work performed on the artifact.

In order to control the evolution of the artifact, each artifact must contain a *lifecycle (model)*.

Definition 0.5 (Artifact Lifecycle Model). The lifecycle model describes the crucial, business-relevant states in which the artifact can be found, as well as rules and constraints governing who, how and when can process the artifact.[3]

The lifecycle model is often formally encoded as a finite state-machine, although other models can be used. It is used to monitor and control the progressing of the artifact through the business process. While the business process owner cannot influence how exactly the process is executed, it can ensure that different artifacts fulfill certain properties at certain times, and with respect to other artifacts, by encoding these expectations and constraints into the lifecycle model. This way, the artifact encapsulates enough information to be able to move through the workflow on its own. Artifact states are used also to monitor the execution of the entire process. At any point during the runtime, the state of the business process is represented by the union of the current states of all the artifacts belonging to the process.

In general case, a single artifact may be changed through different tasks at different times or concurrently. In order to ensure consistency of artifacts' states, these changes need to be performed through transactions.

2.1 Applying Artifact-Centric Incentives in Crowdsourcing Environments

We propose applying the artifact-centric paradigm for defining incentive mechanisms for complex, crowdsourced business processes. Existing incentive mechanisms focus only on the behavior of individuals and teams[23]. The approach we

[2] Adapted from [10]

[3] Adapted from [18]

propose here instead focuses on multiple aspects of human participation in business processes at fine-grained levels. It incorporates the existing personal incentive mechanisms and includes them into the novel incentive model.

The novel artifact-centric incentive model should be viewed as an integral part of the artifact itself. This means that the artifact becomes self-sufficient in human-based workflows, in the sense that the artifact itself attracts and motivates workers to perform targeted actions and to work through the states of the artifact's lifecycle, effectively performing an artifact-driven orchestration.

To help us illustrate the idea better, let us introduce a simple motivating example employing the concept of artifact-centric incentives that we will use in the rest of the paper for identifying and analyzing various requirements for building such systems:

2.1.1 Motivating example

Consider a service that crowdsources building of a simple web page for customers (Fig. 1):

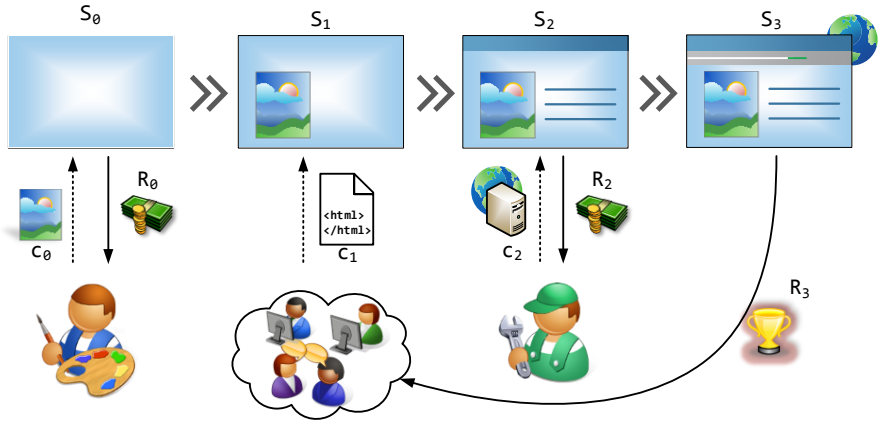

Fig. 1: Artifact-centric representation of a simple software development process. S_i — artifact states. R_i — per-state rewards (incentives). c_i — worker contributions to the artifact's data model.

A customer submits an informal description of web page requirements (Product Requirement Document — PRD). In order to build the web page, a professional is required to discuss the requirements with the customer in detail and to produce an artifact containing functional specification at the technical level (Functional Specification Document — FSD), which must be approved by the customer. Once the FSD is available, a designer can produce the graphics (GR), and a web developer incorporate the graphics with the programming code to produce the html artifact embedding the graphics (HTML). A tester then uploads the web page, tests it a-

gainst the FSD, producing a final report (FR), which must be finally approved by the customer.

To keep the use case simple, let us assume that the FSD contains just three lifecycle states — IN_PROGRESS, CUSTOMER_APPROVED and DEVELOPER_APPROVED. Upon submitting the PRD, the new FSD is created and put into IN_PROGRESS state. An incentive associated with this state is offered, e.g., either a FCFS strategy with monetary reward increasing over time, or a reverse auction, as specified by the customer. The customer also specifies other constraints, such as time constraints for setting the artifact into CUSTOMER_APPROVED state, and the minimal quality metrics of workers (reputation, expertise).

The FSD artifact is then offered in the crowdsourcing market. The system that manages the market does not pick out the workers, but rather limits itself to advertising the task (artifact) to potentially interested candidates — those who are available and fulfill the quality requirements.

Once a worker (requirements engineer) applies and commits to working on the artifact, an activity is created for him, as in [30]. Although the creation of a functional specification document usually requires many activities, iterations, document changes and interactions with the customer, the system will not enforce any particular workflow on the worker, but will rather let him organize it completely to his will. The customer's approval will ultimately allow the FSD artifact to transition into the CUSTOMER_APPROVED state

The FSD now contains some precise graphical requirements and guidelines, out of which a new artifact GR is created. The GR will be used for attracting graphical designers, instructing them, rewarding them, and collecting the produced graphical elements for the web page. It is in the form of a web page, stating the requirements, and promising the reward. A potential incentive strategy here is the tournament reward, where the best designs are awarded, based on the subjective evaluation of the customer[23].

The web developer is chosen similarly to the requirements engineer. The HTML artifact also contains 4 states: AWAIT_FSD_APPROVED, IN_PROGRESS, CUSTOMER_APPROVED, TESTER_APPROVED. Once the developer commits to producing the HTML artifact, he finds the artifact initialized into the AWAIT_FSD_APPROVED state. In order to push the HTML artifact into the IN_PROGRESS state, the developer is required to check the FSD first. If the functional specification is clearly written, and allows him to proceed with writing the code based on it, he sets the FSR into the DEVELOPER_APPROVED state, automatically triggering the transition of the HTML artifact into the IN_PROGRESS state. If the FSD still needs to be improved, the developer resets the FSD into the IN_PROGRESS state, requiring the requirements engineer to work more on it. The remainder of the use case is easy to infer.

2.2 Discussion

Let us first explore how the artifact-centric approach influences modeling of incentives. When the actual monetary reward will be paid to the requirements engineer can depend on many different conditions, and it is exactly the expressive richness of these conditions that makes the artifact-centric incentives so powerful. For example, we may want to specify a much higher reward if the FSD gets developer-approved in the first n iterations. Or, we may want to allow an unlimited number of iterations between the developer and the requirements engineer, but tie the reward amount to a time constraint. Or, most commonly, combine the two incentive mechanisms to promote both speed and excellence.

If the customer expects the FSD to be a big document, the requirements engineer may be incentivized to find and coordinate a small team of helpers that will help speed up the process. The customer can control the number of team members by limiting the number of *roles* that can work on a particular artifact. Different team-incentive mechanisms and reward sharing strategies can be used here — see [23].

The actual payments can be performed after certain state transitions, or only after all the artifacts reach their final states. Furthermore, a deferred team bonus may be promised to all the participants to promote good cooperation between different actors in the process.

As explained in Sec. 4.2, each incentive scheme is vulnerable to the elaborate forms of dysfunctional behavior. In our case, this can be a particular problem, as workers are mostly expected to apply/bid for processing the artifacts themselves, allowing them to coordinate and use different strategies to fool the system. This is why it is very important to foresee and handle these situations. Different methods are deployed to fight this kind of behavior:

- Combination of incentives. If we can foresee the negative application effects of a single incentive mechanism, then we can also construct new incentive mechanism to discourage this kind of behavior.
- Commitment protocols. Offering different commitment protocols restricts workers from maliciously obtaining the benefits on account of other collaborators [7].
- Semi-active worker selection & randomization. This presupposes initially choosing the suitable (reputable) workers, and allowing only them to bid for tasks. Additionally, a non-best bid may be randomly chosen to discourage fixing of prices.
- Sealed-bid auctions. They prevent bidders from seeing the offered prices of others.

This short discussion demonstrates why incentive mechanisms need to be specified at various finer-grained levels, rather than at the business process level only. In fact, we can identify the following dimensions/levels for which incentive mechanisms should be definable:

- **State-dependent incentive mechanisms.** Mechanisms associated with a state of the artifact. The state can be represented not only by a "real" state in the lifecycle model, but also by a combination of values of different metrics, such as: current quality, the number of past contributors, current price, urgency, accuracy, importance, etc.
- **Temporal incentive mechanisms**. Mechanisms conditioning the rewarding action with temporal constraints, e.g., reward may increase as a deadline approaches.
- **Artifact-interdependent incentive mechanisms.** Mechanisms allowing users to specify other artifacts to be processed together/dependently/independently (or in different patterns) with this artifact; or, make the reward payment dependent on the outcome/state of another artifact. These incentives would stimulate the crowd to self-organize and loosely follow the data and control flow we envisaged.
- **Personal incentive mechanisms** on:
  - *Individual level.* Mechanisms targeting individuals, or intended to attract specific types of workers (e.g., experienced, efficient, creative, reputable)
  - *Team level.* Mechanisms designed to promote team efforts on the artifact, e.g., by promising team-based rewards.

3 Requirement Analysis

In Sec. 3.1 we will further analyze these abstraction dimensions, and formulate requirements for designing a novel incentive mechanism model to cover them. This model is meant to extend the conventional lifecycle model of the business artifacts. In Sec. 3.2 we will then introduce and analyze a set of crucial requirements for building and managing sustainable crowdsourcing careers over longer periods of time and different employment providers.

3.1 Requirements for Artifact's Incentive Model

3.1.1 State-dependent Incentive Mechanisms.

Finite state machines are the most commonly used formalism to describe the lifecycle model of an artifact due to their expressiveness, comprehensible semantics and tool support. Consequently, it comes natural to use artifact states in conditions for applying incentive mechanisms. However, since most artifacts are documents intended to be processed by humans, their lifecycle models need to be kept reasonably simple. This means that we often lack the fine granularity needed for expressing an incentive condition.

This is why we propose that, apart from the artifact's regular lifecycle states visible to humans and used for guiding the business process — *hard states*, a set of machine-processable *soft states* for regulating incentives also be specified. Soft states could be defined as sub- or super-states of existing states that contain no entry or exit transitions, but are 'entered' whenever the lifecycle model is in the associated hard state and the *entry predicate* for the soft state holds true.

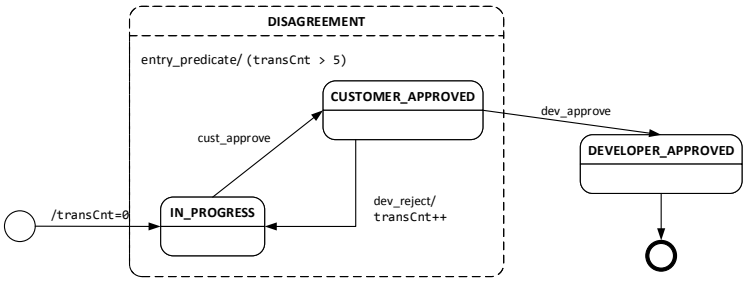

Fig. 2: State-dependent incentives. Soft states are outlined with the dashed line.

Entry predicates would allow us to specify various metric thresholds as conditions for applying an incentive mechanism. In our motivation example, this would allow us to introduce a metric *transCnt* that would keep count of the number of transitions between CUSTOMER_APPROVED and IN_PROGRESS hard-states of the FSD artifact. A super-softstate DISAGREEMENT, comprising both CUSTOMER_APPROVED and IN_PROGRESS can be defined with the entry predicate: $transCnt > 5$ (Fig. 2). Detecting that there is a disagreement on the functional specification between the requirements engineer and the developer is an important fact to consider when deciding which incentive mechanisms to apply. In our case, entering the DISAGREEMENT state could be used as signal for applying an incentive mechanism that will help resolve the issue, e.g., by promising a penalty if the agreement is not met in a specified time, or by discontinuing the engagement of the workers.

Of course, incentive conditions could be specified just as predicates for the purpose of constructing incentive mechanisms, i.e., without introducing the notion of soft states. However, conceptualizing the conditions as states and associating them with hard states forces incentive designers to use the artifact-centric paradigm, reducing the number of possible conditions and making them addressable entities in the model. Also, in order to exhibit effect, certain incentive mechanisms must be presented in advance to the workers. In these situations it is helpful to have a limited number of incentive conditions associated with artifact states, making the incentives transparent and understandable to workers.

However, the main advantage of this approach is separation of concerns; while an artifact's hard states can be standardized for use throughout different companies, company-specific soft states can be defined to support specific incentives and applied to existing artifact lifecycle models without affecting their primary usage.

3.1.2 Temporal Incentive Mechanisms.

Including temporal dimension in the artifact's lifecycle model is essential, as incentive mechanisms exhibit their effects only if promised in advance and applied upon an action is completed. Furthermore, it is essential to be able to encode proper/expected ordering of events leading to a reward or punishment, or to detect activities taking too much time. Therefore, the time management must include both time-interval semantics, as well as the event ordering.

A way to meet these requirements would be incorporating the time model and the operators of the Linear Temporal Logic (LTL). LTL operates on a simple, discrete, linear time model, isomorphic to the set of natural numbers $\mathbb{N}$. The time moments (*ticks*) are therefore counted from the agreed 'beginning of time' onwards. The events happen at ticks. Events and states are represented by logical propositions that can treated with a set of temporal and standard logical operators.

While any platform-specific implementation of incentive mechanisms must include time queries in some way, to the best of our knowledge, there are no known systematic approaches to modeling temporal logic operators in the domain of incentive management. In the area of BPM, on the other hand, we have seen successfull attempts of including LTL into process models [19].

We propose introducing declarative LTL constructs for incentive mechanisms *on the artifact level* by applying similar principles as in [19]. The LTL constructs can be used to express temporal propositions for various incentive conditions. For instance, looking back at our example, we can specify that *after* the HTML artifact gets into the TESTER_APPROVED state, a BUG_REPORT artifact should *never* be approved for a missing feature. Of course, in real systems, 'never' will have a limited duration, after which the whole proposition should expire, e.g., after an iteration's end (Fig. 3).

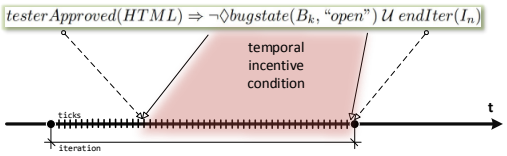

Fig. 3: A temporal incentive condition encoded in LTL.

Another beneficial notion we suggest be introduced into the artifact lifecycle model is the notion of *iterations*. Iterations are time intervals with just-in-time initialization and finalization. They can be used for representing work phases meaningful to humans that are inherently unstable, such as sick leave, working hours or project phase. This means that we could define an iteration for the purpose of describing that phase. However, when designing an incentive mechanism, we may not know exactly when the iteration would start, nor when it would end. Therefore, we would express the incentive conditions by using iterations, rather than ticks, and

leave it to the underlying system to signal the iteration's starting and ending times and handle the incentive execution. The iteration abstraction can be expressed in LTL, but we suggest using it along with the standard LTL operators for simplifying the time management as it corresponds better to the organization of human work.

Including declarative LTL constructs on the artifact level adds a new dimension of expressiveness to the incentive mechanisms. In addition, the constructs can be used for runtime monitoring of crowdsourcing platforms for specifying temporal invariants.

3.1.3 Artifact-interdependent Incentive Mechanisms.

In complex collaborative efforts, such as software development processes, the lifecycle of a single artifact cannot be considered independently of the states of other artifacts in the business process. Therefore, it becomes imperative to formally capture these dependencies in the lifecycle model.

We propose formalizing the dependencies among different artifacts so that they can be used to express different incentive conditions. These conditions can then be integrated into the incentive model used to augment the lifecycle model of the artifact. To the best of our knowledge, the concept of relating the lifecycle states of different artifacts to express incentive conditions has never been formally proposed before.

The paper [14] presents one possible formalism that could be adapted for such a purpose. It enriches the conventional artifact lifecycle model by introducing the notion of 'state contexts' and 'context-aware state transitions'. The contexts are defined graphically. Relationships between artifact and role entities in context definition offer the expressiveness of the first-order predicate calculus. This allows us to express the necessary artifact interdependencies. Figure 4 (borrowed from [14]) displays an example of the graphical notation. For more information, the reader is referred to the original paper.

For example, we could use this notation to express that the FSR needs to be moved into the DEVELOPER_APPROVED state first in order for the HTML artifact to move into the IN_PROGRESS state. But we can also use the same notation to, for example, prevent a reward being paid if there is at least one bug report in the unresolved state. The benefit of using a graphical notation that includes universal and existential quantifiers is that it makes it easy for humans to specify and reuse this type of conditions.

3.1.4 Personal Incentive Mechanisms.

As each personality is different, a single incentive can never work the same way on every person. The conventional approach when designing the incentives for a particular crowd effort is to select those that suite best an average worker in the targeted group. However, unless this group is large enough this approach may not

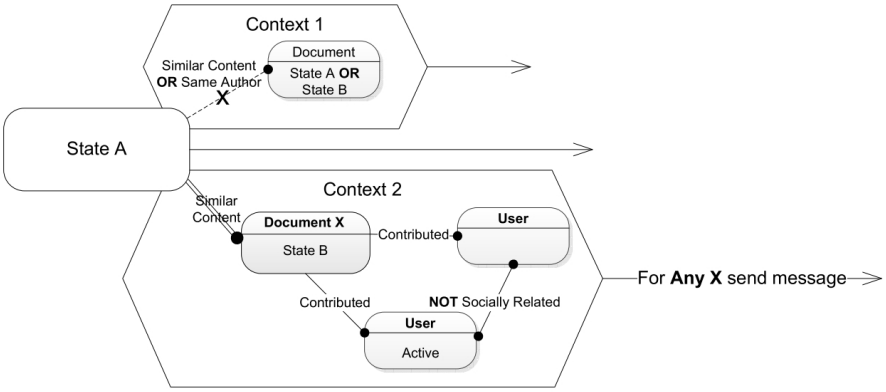

Fig. 4: An example of artifact interdependency contexts (from [14]).

perform well. Indeed, in limited efforts, just a few participants with a particular interest or affinity for that task may contribute much more than the rest of the crowd [21]. Therefore, for assembling small-scale teams focused on specific tasks/artifacts it is important to identify and attract such individuals. One way of achieving this is through personalized incentives.

For example, if we want to attract a promising, young software engineer to our team, then we cannot expect him to be able to solve certain tasks as fast as an engineer already experienced in that area. That is why we may be willing to value and reward his effort levels rather than his speed. We may also tolerate certain errors (e.g, failed code reviews, reopened bugs) and not penalize him, hoping to improve his engineering skills for future collaborations. On the other hand, employing an experienced senior engineer implies paying him more, but also evaluating his performance on speed and quality metrics.

Therefore, the artifact's incentive model should offer different "incentive packages" (Fig. 5) that consider different metrics and promise different rewards appealing to different groups of workers. Incentive packages are a way of including existing research on modeling personal incentives (see [23]) into the new artifact-centric paradigm. It should be possible to enable/disable incentive packages as needed, e.g., when enough workers apply for one incentive package, or when the reward money runs out. Also, it should be possible to specify inter-package enabling conditions — e.g., requiring a number of workers (non-)applying for another package first (or at the same time). For instance, an incentive package targeting a team lead should be enabled only if the package meant to attract developers managed to attract a sufficient number of appropriate candidates.

Incentive packages could in special cases target particular individuals rather than groups. In this case, we can rely on the particular worker's behavioral history to infer (by machine learning) potentially interesting activities, tasks and collaborators to the worker. If the artifact's lifecycle model foresees a potentially favorable set of conditions that could attract this particular worker, then a tailored incentive can be

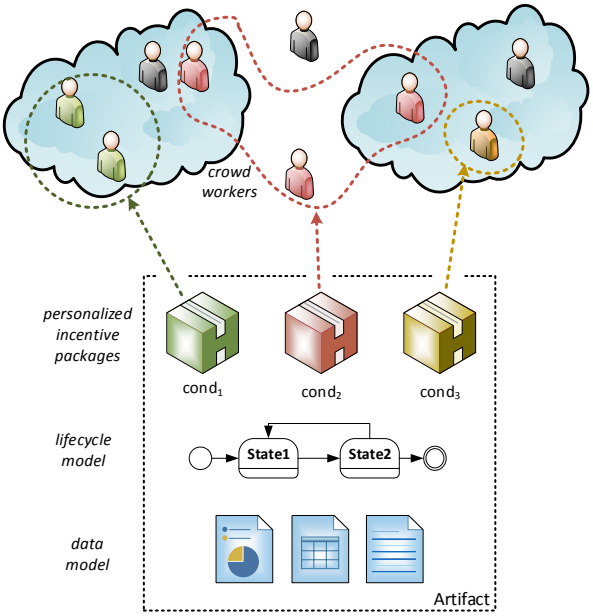

Fig. 5: Personalized incentives help attract (groups of) workers with specific properties.

offered to attract him to work on the artifact. Multiple individuals could be targeted in parallel, but each worker could only claim the reward of his personal incentive package.

An interesting example where this approach could be successfully used are the so-called *structural incentives*, i.e., incentives that motivate people by promising to establish certain social or professional collaboration relationships/patterns between workers. For example, young professionals may find the possibility to collaborate with renowned experts to be more attractive in a short term than a higher salary because of the prestige associated with it. Similarly, the possibility to collaborate with known and trusted collaborators from the past[25] may be the determining factor in choosing to work on one artifact over another.

For example, by analyzing the code repository logs we could determine that developer A often collaborated with developers B and C on the same .java files, and that they were often reviewing each other's code submissions. Based on the code snippets they were submitting, we can infer their common expertise, e.g, which databases or libraries they used [26]. This gives us reason to believe that the same three persons collaborating on a new project within their area of expertise are probably going to be productive. For this reason, we may want to incentivize them to join our effort, and put out three individual incentive packages targeting them. The incentive for developer A could contain the condition that at least one of the other

two developers would also have to accept working on the artifact. The packages for developers B and C would be similar.

It is probable that the developers A, B and C would more likely join an effort with known collaborators. Therefore, the application of multiple personalized incentives can also exhibit a significant group effect, while transferring the organizational and motivational burden onto workers themselves, since they would be persuading each other much more efficiently than an automated system could do.

While personal incentives can achieve powerful motivating effects, their expressiveness and limitations fully depend on the adopted underlying model of personal incentives. However, rather than discussing the properties and limitations of the different existing personal incentive models, here we limit ourselves to suggesting how the existing models can be integrated into the encompassing artifact-centric incentive model.

3.2 Requirements for Sustainable Crowdsourcing Careers

One of the biggest problems when dealing with incentives in crowdsourcing in general (and especially with personal incentives) is selecting and defining metrics to accurately describe current worker contributions and appropriately interpret past performance in the current context. Solving this problem would in theory allow different employers to track and update the performance history of the crowd workers in a uniform way, and allow the workers to use the reputation records with different employers very much like CVs and recommendation letters are used today. We call this (temporal and locational) *transfer of reputation*. The notion of transferable reputation is one of the key enabling conditions for successful application of personal incentives.

Unfortunately, defining a comprehensive set of metrics to cover so many aspects of human work that would allow us to build a uniform record of one's working history is impossible. Even though, for certain highly-specific domains, it may be possible to define referent metrics ontologies, in majority of real-life applications this is not a viable solution, nor one that will likely get embraced by the employers. Furthermore, a metric's relevance may change with time.

This is why we suggest not to predefine specific metrics to be kept, but rather keep a public history of worker's performance and employ a *reputation service* for *just-in-time metric assessment* as a cost-effective alternative to the development of dedicated metrics. In this way, the ad-hoc invoked service would map a worker's performance records spanning a specified time period into a set of given, context-specific metrics of interest to the current employer.

Different reputation service providers could offer different QoS at different prices, according to the needs of the employer (Fig. 6). For example, for performing a simple programming task, the employer may require "someone with basic programming skills". This means that the reputation service needs to return a metric indicating whether a candidate has done programming before. A software web ser-

vice that will check the candidate's activity metrics on sites such as StackOverflow, or recommendations on sites such as LinkedIn can be employed here to return/calculate a rough estimate of the worker's reputation. However, the service will produce results immediately, and will cost little. On the other hand, if the employer needs "an Informix database security expert" in his team, then the employer may want to use a human-based service (HBS) [24] employing *subjective evaluations*[23] from other software developers who would be asked to review the candidate's personal work history or even his code from open-source projects. Better QoS, though, would probably imply longer invocation times and higher price. Invocation results should be appended to the existing worker's history, and serve as another piece of data valuable for future evaluations, especially for monitoring the development of worker's skills and working attitude.

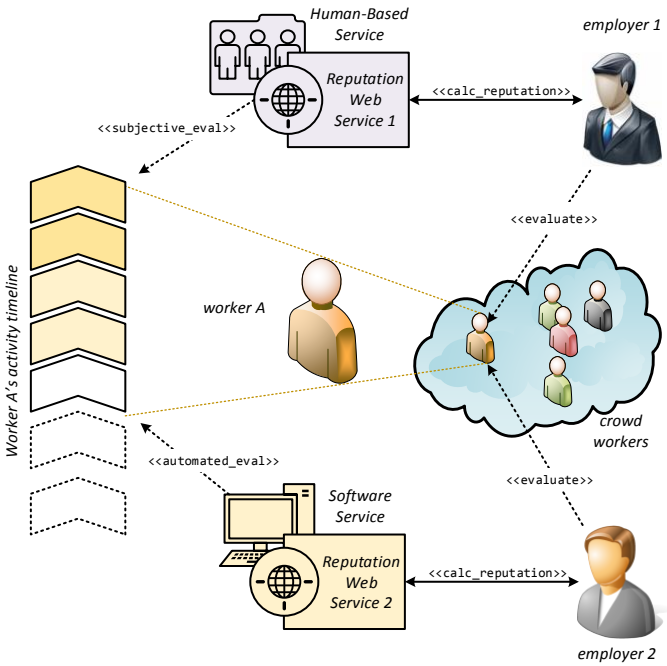

Fig. 6: Reputation is evaluated from worker's public history records and interpreted upon request through reputation service.

This approach would allow employer to keep using any internal labor metrics he wants, while allowing transfer of reputation through shared activity history whose meaning is mapped to particular metrics via the reputation service.

3.3 Requirements Summary

In previous sections we explored the different aspects we find worth of including in a future incentive model for socio-technical/crowdsourcing systems. We presented suggestions in form of requirements, providing simple, but illustrative examples as justification, and discussing potential issues and benefits. Table 1 presents a high-level summary of this requirement analysis. Although non-exhaustive, it provides a useful overview of the different levels at which incentives need to be addressed, opening up space for more focused research.

	Application level	Motivation	Proposed requirements
Incentive model level	artifact state	defining per-state incentive mechanisms	incentive conditions as soft states
	time	expressing temporal incentive conditions	time-interval semantics and event ordering
		scheduling of deferred rewarding actions	iterations
		evaluating/mining past work	
	inter-artifact	allowing the states of other artifacts influence incentives offered for processing this artifact	formalism for expressing artifact inter-dependencies
	worker(s)	attracting particular individuals to work on artifact	personalized incentive packages
			inter-package dependencies
		transferring organizational effort to humans	structural incentives
Inter-organizational level	spanning multiple crowdsourcing employers	enabling transfer of worker reputation between employers	public record of worker performance
		creating favorable conditions for sustainable “careers in the cloud”	ad-hoc, (human-) service-based interpretation of the metrics in current context

Table 1: Summary of requirements for supporting artifact-centric incentive mechanisms in crowdsourcing environments.

4 Related Work

4.1 Artifact-centric Business Process Modeling

Artifact-centric BPM, also known as 'document-centric' or 'data-driven' BPM has attracted a lot of research attention in the past. Here we will review only a small selection of fundamental papers that enable the reader to understand the background and motivation for our approach.

One of the landmark ideas of the artifact-centric paradigm is that it is possible to design workflow systems without explicit control flow, where the actual execution is governed by the artifacts themselves, also serving as input as outputs. The paper [30] presets a prototype implementation of a document-driven workflow system, demonstrating the feasibility of this approach. In [18] the authors informally describe the business artifact concept and its lifecycle models, while [1] introduces a formal model and operational semantics. Authors of [8] analyze the problem of verification of artifact behavior in operational models. The paper [16] presents a methodology and patterns for building up real business operational models using artifacts. Finally, [10] presents a comprehensive survey of the fundamental research on artifact-centric BPM.

For an overview of more recent developments in the area, the reader is referred to the following publications: [3, 4, 6, 11, 28, 29]

4.2 Incentives & Rewarding

Related work in the area of incentives originates mostly from economics, game theory, organizational science and psychology. The principal economic theory treating incentives today is the *Agency Theory* [2, 13]. Incentives are defined as the principal mechanism for aligning interests of business owners and workers. As a single incentive always targets a specific behavior and induces unwanted responses from workers [13], multiple incentives are usually combined to counteract the dysfunctional behavior and produce wanted results. Opportunities for dysfunctional behavior increase with the complexity of labor, and so does the need to use and combine multiple incentives. The paper [20] presents a comprehensive review and comparison of different incentive strategies in traditional businesses.

The number of computer science papers treating these topics is limited. Incentives are discussed usually within particular, application-specific contexts, like peer-to-peer networks, agent-based systems and human-labor platforms (e.g., Amazon Mechanical Turk), rather than being considered at a general level. In [22] the aim is to introduce appropriate incentives to maximize peer-to-peer content sharing. In [31] the authors seek to maximize the extent of social network by motivating people to invite others to visit more content. In [15] the authors try to determine quality of crowdsourced work when a task is done iteratively compared to when it is done

in parallel. In [17] the authors investigate how different monetary rewards influence the productivity of mTurkers. In [21] the authors compare the effects of lottery incentive and competitive rankings in a collaborative mapping environment. In [9] the authors analyze two commonly used approaches to detect cheating and properly validate submitted tasks on popular crowdsourcing platforms.

An overview of typical incentives and rewarding practices in crowdsourcing systems can be found in [23, 27]. A common conclusion is that incentives used in today's social computing platforms are mostly limited to simple piece-rates that may be suited for simple task processing, but are inappropriate for the more advanced collaborative efforts such as software development. However, both studies suggest that, depending on the environment, there exist appropriate types of incentives that combined together should succeed in motivating and rewarding workers taking part in more complex or intellectually more challenging labor activities.

5 Conclusion

We believe that, in order to support collaborative processes of increased complexity, the crowdsourcing platforms will need to leverage human-based services to tackle important challenges such as team formation, adaptability and runtime management of collaboration processes. However, introducing humans into the loop requires specific methods for attracting, motivating and controlling humans. We suggested this could be done with a combination of artifact-centric workflows and rich incentive mechanisms. We then analyzed different aspects that an incentive mechanism model for such systems should cover, and suggested integrating it into the artifact lifecycle model to create encapsulated units that can be offered to the crowd for processing. The novel artifact model would allow the crowd workers to independently drive the processing in the envisioned direction and tackle the aforementioned challenges. The result of our analysis is a set of requirements that the future systems should support, ultimately providing a working environment that would promote fairness, worker's reputation transfer and ultimately, a fundamental step towards building a framework for managing "careers in the cloud".

References

[1] Bhattacharya, K., Gerede, C., Hull, R., Liu, R., Su, J.: Towards formal analysis of artifact-centric business process models. In: Business Process Management, pp. 288–304. Springer Berlin Heidelberg (2007). DOI 10.1007/978-3-540-75183-0_21

[2] Bloom, M., Milkovich, G.: The relationship between risk, incentive pay, and organizational performance. The Academy of Management Journal **41**(3), 283–297 (1998)

[3] Cohn, D., Hull, R.: Business artifacts: A data-centric approach to modeling business operations and processes. IEEE Data Eng. Bull. **32**(3), 3–9 (2009)
[4] Damaggio, E., Hull, R., Vaculín, R.: On the equivalence of incremental and fixpoint semantics for business artifacts with Guard-Stage-Milestone lifecycles. Information Systems **38**(4), 561–584 (2013). DOI 10.1016/j.is.2012.09.002
[5] Dustdar, S., Truong, H.l.: Virtualizing software and humans for elastic processes in multiple clouds – a service management perspective. International Journal of Next-Generation Computing **3**(2) (2012)
[6] Fritz, C., Hull, R., Su, J.: Automatic construction of simple artifact-based business processes. Intl. Conf. on Database Theory (ICDT '09) p. 225 (2009). DOI 10.1145/1514894.1514922
[7] Gal, Y., Grosz, B., Kraus, S., Pfeffer, A., Shieber, S.: Agent decision-making in open mixed networks. Artificial Intelligence **174**(18), 1460–1480 (2010). DOI 10.1016/j.artint.2010.09.002
[8] Gerede, C., Su, J.: Specification and verification of artifact behaviors in business process models. Intl. Conf. Service-Oriented Computing (ICSOC 2007) pp. 181–192 (2007)
[9] Hirth, M., Hossfeld, T., Tran-Gia, P.: Analyzing costs and accuracy of validation mechanisms for crowdsourcing platforms. Mathematical and Computer Modelling (2011) (2012). DOI 10.1016/j.mcm.2012.01.006
[10] Hull, R.: Artifact-centric business process models: Brief survey of research results and challenges. In: On the Move to Meaningful Internet Systems (OTM), pp. 1152–1163 (2008)
[11] Hull, R.: Towards Flexible Service Interoperation Using Business Artifacts. 15th IEEE Intl. Conf. Enterprise Distributed Object Computing (EDOC) pp. 20–21 (2011). DOI 10.1109/EDOC.2011.27
[12] Kaganer, E., Carmel, E., Hirschheim, R., Olsen, T.: Managing the human cloud. MIT Sloan Management Review **54**(2), 22–32 (2013)
[13] Laffont, J.J., Martimort, D.: The Theory of Incentives. Princeton University Press, New Jersey (2002)
[14] Liptchinsky, V., Khazankin, R.: A novel approach to modeling context-aware and social collaboration processes. In: 24th Intl. Conf. Advanced Information Systems Engineering (CAiSE'12). Springer, Gdansk, Poland (2012). DOI 10.1007/978-3-642-31095-9_37
[15] Little, G., Chilton, L.B., Goldman, M., Miller, R.C.: Exploring iterative and parallel human computation processes. In: Proc. ACM SIGKDD Workshop on Human Computation, HCOMP '10, pp. 68–76. ACM, New York, NY, USA (2010). DOI 10.1145/1837885.1837907
[16] Liu, R., Bhattacharya, K., Wu, F.: Modeling business contexture and behavior using business artifacts. In: J. Krogstie, A. Opdahl, G. Sindre (eds.) Advanced Information Systems Engineering, *Lecture Notes in Computer Science*, vol. 4495, pp. 324–339. Springer (2007). DOI 10.1007/978-3-540-72988-4_23
[17] Mason, W., Watts, D.J.: Financial incentives and the performance of crowds. In: Proc. ACM SIGKDD Workshop on Human Computation (HCOMP

'09), vol. 11, pp. 77–85. ACM, Paris, France (2009). DOI 10.1145/1600150.1600175

[18] Nandi, P., Kumaran, S.: Adaptive business objects - a new component model for business integration. In: Proc. 7th Intl. Conf. Enterprise Information Systems (ICEIS '07), pp. 179–188. Miami, USA (2005)

[19] Pesic, M., Schonenberg, H., van der Aalst, W.M.: DECLARE: Full Support for Loosely-Structured Processes. In: 11th IEEE Intl. Conf. Enterprise Distributed Object Computing (EDOC '07), pp. 287–287. IEEE (2007). DOI 10.1109/EDOC.2007.14

[20] Prendergast, C.: The provision of incentives in firms. Journal of economic literature **37**(1), 7–63 (1999). URL http://www.jstor.org/stable/2564725

[21] Ramchurn, S., Huynh, T., Venanzi, M., Shi, B.: Collabmap: crowdsourcing maps for emergency planning. In: Proc. ACM Web Science. Paris, France (2013). URL http://eprints.soton.ac.uk/350677/

[22] Sato, K., Hashimoto, R., Yoshino, M., Shinkuma, R., Takahashi, T.: Incentive mechanism considering variety of user cost in p2p content sharing. In: Global Telecommunications Conference (IEEE GLOBECOM '08), pp. 1–5. IEEE (2008). DOI 10.1109/GLOCOM.2008.ECP.426

[23] Scekic, O., Truong, H.L., Dustdar, S.: Incentives and rewarding in social computing. Communications of the ACM **56**(6), 72 (2013). DOI 10.1145/2461256.2461275

[24] Schall, D., Dustdar, S., Blake, M.B.: Programming Human and Software-Based Web Services. Computer **43**(7), 82–85 (2010). DOI 10.1109/MC.2010.205

[25] Schall, D., Skopik, F., Psaier, H., Dustdar, S.: Bridging Socially-Enhanced Virtual Communities. In: ACM SAC 2011 (2011)

[26] Teyton, C., Falleri, J.R., Blanc, X.: Mining Library Migration Graphs. In: 19th Conf. on Reverse Engineering, pp. 289–298. IEEE (2012). DOI 10.1109/WCRE.2012.38

[27] Tokarchuk, O., Cuel, R., Zamarian, M.: Analyzing crowd labor and designing incentives for humans in the loop. Internet Computing, IEEE **16**(5), 45–51 (2012). DOI 10.1109/MIC.2012.66

[28] Vaculin, R., Heath, T., Hull, R.: Data-centric Web Services Based on Business Artifacts. 19th Intl. Conf. on Web Services (ICWS '12) (1), 42–49 (2012). DOI 10.1109/ICWS.2012.101

[29] Vaculin, R., Hull, R., Heath, T., Cochran, C., Nigam, A., Sukaviriya, P.: Declarative business artifact centric modeling of decision and knowledge intensive business processes. In: 15th Intl. Conf. Enterprise Distributed Object Computing (EDOC '11), pp. 151–160. IEEE (2011). DOI 10.1109/EDOC.2011.36

[30] Wang, J., Kumar, A.: A framework for document–driven workflow systems. In: W.M. van der Aalst, B. Benatallah, F. Casati, F. Curbera (eds.) Proc. Intl. Conf. Business Process Management (BPM '05), *Lecture Notes in Computer Science*, vol. 3649, pp. 285–301. Springer (2005). DOI 10.1007/11538394_19

[31] Yogo, K., Shinkuma, R., Takahashi, T., Konishi, T., Itaya, S., Doi, S., Yamada, K.: Differentiated Incentive Rewarding for Social Networking Services pp. 169–172 (2010). DOI 10.1109/SAINT.2010.65