International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 4 Issue: 2 204 - 213

CrowdCE: A Collaboration Model for Crowdsourcing Software with Computing Elements

Tarek Ali, Mervat Gheith
Department of Computer and Information Sciences,
Institute of Statistical Studies and Research,
Cairo University, Egypt
{tarekmmmmt, mervat_gheith}@yahoo.com

Eman S. Nasr
Independent Researcher
Cairo, Egypt
nasr.eman.s@gmail.com

***Abstract*—**Today's crowd computing models are mainly used for handling independent tasks with simplistic collaboration and coordination through business workflows. However, the software development processes are complex, intellectually and organizationally challenging business models. We present a model for software development that addresses key challenges. It is designed for the crowd in the development of a social application. Our model presents an approach to structurally decompose the overall computing element into atomic machine-based computing elements and human-based computing elements such that the elements can complement each other independently and socially by the crowd. We evaluate our approach by developing a business application through crowd work. We compare our model with the traditional software development models. The primary result was completed well for empowering the crowd.

Keywords-*Decomposition and software development model, distribution and social computing model, computing element, crowdsourcing software, crowd work, task management.*

*****

I. INTRODUCTION

Machine-Based Computing Element (MBCE) is an unprecedented power to transmit information over large distances, to store and manipulate data for a long time and to quickly perform well-defined formal computations. *Human-Based Computing Element (HBCE)* is a human ability for computation to solve problems that are trivial for humans, but complex for machines. It depends on competencies, knowledge, and skills with networks of social relationships and understanding of social context. For example, in complex work labeling annotating, cleansing, evaluating data/content, detecting patterns, classifying objects, and steering analytics [*1*]. *Computing Element (CE)* is a computational element that blends both MBCE and HBCE in a hybrid class. This class can be deployed and utilized as a collective on-demand based on a different quality, cost, time and incentive models. For example, Amazon Mechanical Turk, wikis, CAPTCHA [*2*], reCAPTCHA [*3*], KA-CAPTCHA [*4*], WS-HumanTask, BPEL4People and GWAP. Zhang [*5*]. From the *Distributed Computing* point of view, a problem is divided into many tasks, each of which is solved by one processor (node). Each processor has its own private (local) memory (distributed memory). Finally, the information is exchanged by passing messages between the processors using the available communication links. Fig.1 is self-explanatory and it demonstrates the root of our area.

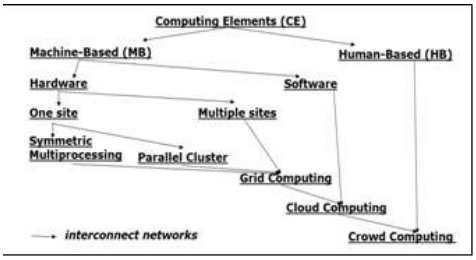

Figure 1. Distributed computing and computing architectures.

Socially Intelligent Computing (SIC) is a newly emerging field that refers to the recent efforts on the modeling crowd computing to understand the ways in which systems of human intelligence across the globe and social networks can work together as efficiently as a giant machine [*6*]. If we were thinking in this way using this metaphor, we can define CrowdCE as it is a modelfor managing the collaboration and coordination of MBCE and HBCE. It unifies humans and software; and supports ad hoc and process-centric collaborations. This is done bydecomposing, developing and then composing collaborative CEs.

Crowd labor has a number of potential benefits over the existing approaches [*7*]. In this paper, we present a new collaboration model for use in the *Software Development Life Cycle (SLDC)* where the development phases are done by crowd labor. However, existing crowd computing models have styles that make them difficult to cover all types of crowd work. Those difficulties are related to task complexity and task management. Among other issues, crowd computing at scale requires a systematic model to decompose an application's CE into atomic MBCE and HBCE, and then efficiently composing the results into a hybrid-CE. This hybridity is suitable for distribution to the crowd and machine.Moreover, it can be completed socially within a short amount of time. It is largely independent and the response to each CE can be automatically tested and integrated to form the overall development phases.

The development process is as follows. a) Post requirements business model as HBCE. MBCE builds technical architects using *decomposition and software development model* [*8*] [*9*]. b) Components of the HBCE and MBCE are specified as Hybrid-CE consisting of interfaces, test cases, and a textual description. Distribution and social computing model [*10*] [*11*] are leveraged here. c) HBCE is created for the crowd to execute the task to fillfile

requirements. d) As the crowd submits the work, automatic verification is done using the knowledge base system (test cases). All results are assembled even with a subset of the crowd work. To validate our model, we conducted an experiment of developing a business application using crowd labor. The design of the application, the MBCE and HBCE were created adhering to our model. We posted tasks with different levels of skill tothe crowd to achieve. Because the tasks were different, and according to skilllevel, 100% of the participants were successful. The primary result of the experiment is the validation of the CrowdCE as a mechanism for the independent development of software phases using the different level of skills, and comparing with the traditional System Development Life Cycle (SDLC), the experiment was completed well for empowering the crowd.

The paper is structured as follows. Section II presents the design requirements and usage context in crowd-based software development. Section III presents basic elements of CrowdCE. Our CrowdCE is presented in Section IV. Evaluation and experimental results are presented in Section V. Related worksare discussed in Section VI and we conclude in Section VII.

DESIGN REQUIREMENTS AND USAGE CONTEXT

Design requirements

CrowdCE is an SDLCmodel intended for use by crowds. It tries to align their work and provides them the following functionalities as a set of basic design requirements such as supporting the high-level decomposition and composition; high-level representation techniques, execution environment, workflow business management functionality; collaboration environment, transparency working conditions, andsimplification a complex work to cover all levels of skills.

Usage context

Human registers his/her profiles (such as enlisting for performing different simple or professional activities). Using human's profiles for locating and engaging different crowd into different collaborative efforts, under the working conditions and their needs. The CrowdCE asks for an explicit approval from crowd under a short-term contractual relationship. The application-specific business logic, development life cycle and accepted quality of result are encapsulated. Also, different adaptation, integration mechanism, metrics and other parameters are presented at runtime execution, and during the CE execution, various incentives might be applied [*12*].

BASIC ELEMENTS OF CE

The role of architecture gives us a general view as Fig. 2a shows, an MBCE interacts with HBCE receiving state information and reinforcement feedback and executing actions. The state information may depend on each other. That is, the next state may depend on current state and on the executed action. Moreover, this architecture is thus more difficult, but also more natural, simulating the interaction with an actual outside world. The table in Fig. 2b describes four categories of CE. Each of these categories relies on a Machine(M) or Human(H). The CE is in terms of the roles (innovation (initiation) or selection (decision)) performed in each case by M or H through computational processes (P). These categories of CE can be referred by two-letter abbreviations: HM, MH, HH, and MM. Here the first letter identifies the type of CE performing innovation, the second letter specifies the type of selection CS. In some implementations.

The following examples demonstratethe four categories in developing complex applications. *HH*: Such as Wiki enabled designing the web content by multiple users, i.e. supported two types of H-based innovation (contributing new design and its incremental improvement). However, it might be a tool supporting collaborative content evolution. A human-basedgenetic algorithm which uses both HBCE in terms of selection and three types of human-based innovation (contributing new content, mutation, and recombination). Thus, all HBCEs of a typical genetic algorithm are outsourced to humans e.g. integrating crowds with a genetic algorithm to study creativity. Collaborative filtering in social search applications, these applications accept contributions from the crowd and attempt to use human evaluation to select the fittest contributions that get to the top of the list. *HM:* Such as the games where several programs written by people compete in a tournament. Actors of the programs copy, modify and recombine successful strategies to improve their chances of winning. In Computerized tests, M generates a problem and search for HBCE. E.g., CAPTCHA tells human users from computer programs by presenting a problem that is supposedly easy for a human and difficult for a computer. It is useful where scanning old books that optical character recognition cannot decipher. In Interactive online games, programs extract knowledge from people in an entertaining way. Such as human swarming or social swarming, real-time closed-loop systems work around groups of networked users molded after biological swarms, enabling human participants to behave asunified collective intelligence. *MH:* Mainly, this category is based on Human-Computer Interaction (HCI) that enables the actor to create an abstract drawing only by selecting his/her favorite images, so human only performs the selection, which is easier for humans; and software performs the innovative role. *MM:*Such as domain knowledge-based and automatic workflow generation.

Effective supports to CrowdCE require generic classification of basic elements for CE (MBCE and HBCE), Actors (X), Activity (A), development Phase (P), and their relationships, but it is not enough. There is a need to know how specific X accomplish their A and what they are doing at particular P, and to provide ways of evaluating how work practices are transmitted in time. In the following section, we discuss the high level of basic elements, which we believe will provide additional support to designCrowdCE, such as

$$CE = \{MBCE, HBCE\} \quad (1)$$

Where *MBCE and HBCE*complement each other. For example, it can be an expert annotation on NLP, generating speech corpora for language research *(*word-sense disambiguation*), the annotating object in the image, identifying depth* layers, solve algorithmic problem (graph coloring), active learning, and semi-supervised learning [*13*].

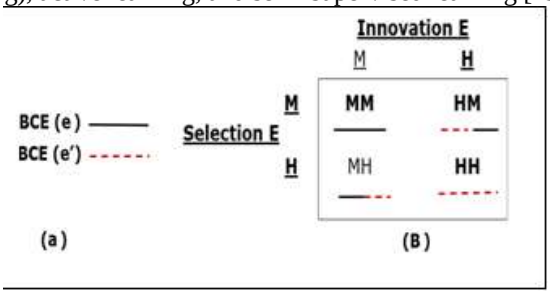

Figure 2. a) Basic component, and b) Basic architecture for xomputing element (E)

Actor (X) over CE

The X model is known for concurrent and distributed computing as a concurrent object-based computation model. The basic element of concurrency is called X. They communicate over asynchronous message passing. Each X has a unique identifier, CE, and an unbounded message queue. There are*Triple actors =*X_{CE}such as

$$X = \{x1, x2, x3\} \quad (2)$$

Where Initiator is x_1, Participants is x_2, and Social Platform is x_3 (see Fig. 3-a).*The initiator (x1)* might be a crowdsourcer, requester, entrepreneur, or event coordinators who initiate the work that has to be developed. Moreover, he/she is a member in x2. *Participants* (x_2) might crowds, performers, or providers who perform the development of task that requiresa certain formal education and domain background to develop the solutions (product or service). *Social Platform* (x_3) might be a technological form or a marketplace where x1 and x2 can meet and communicate to develop specific work, it is computationally-enabled both human and machine.

Activity (A) over CE

It is might micro-taskor action through an SDLC. From (1) and (2), we can deduce such as double Activities=A_{CE}such as

$$A = f(X, CE, C) \quad (3)$$

And $A_{x1}$and $A_{x2}$are HBCEsand A_{x3} is MBCEs, C is a Constraint and each A has a unique identifier, CE, and an unbounded A queue that work together (see Fig. 3-b**)**.For example, X can develop a method to plan, e.g., deciding what is to be done, organize e.g., making arrangements, staff e.g., select the right x2 for the task, monitor e.g., checking on progress, and controlling e.g., taking action to remedy holdups.

Development Phase (P) over CE

P is a collection of A. It is highly flexible; it converts input (resource) to output (product). P is started when x_1 initiates work. P as a parent can initiate another P, which is a called a child P. Ps can communicate and collaborate through exchanging information or synchronize their work. From an SDLC perspective, we can state P such as one or more Phase=P_{CE}such as

$$P = \{p1, p2, \ldots, pn\} \quad (4)$$

And p has a unique identifier and an unbounded A(s) queue that complements each other(see Fig.3-c).For example, decomposition phase is huge work can be split up (decomposed) small, simple and similar pieces that can be developed in parallel without a problem. At the same time, these pieces can be easily decomposed [*14*]. This is can be done through collections of similar or different activities-designing workflow. The same thing for assigning work, and co-ordination e.g., managing the dependencies between activities "parts of small work"

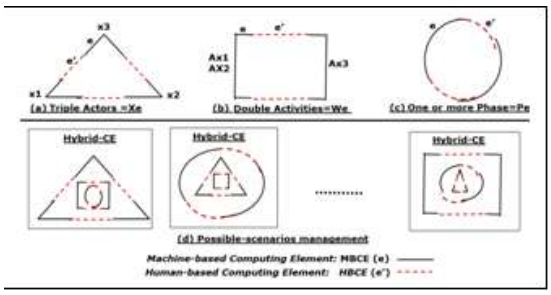

Figure 3.

Figure 4. Unifying CE in crowd collaborations.

Crowd Collaborations over Unifying CE

CrowdCE based on hybrid-CEthat has started exploiting the knowledge thatis increasingly found in the crowd and machine. Itis modeled to understand the CEs whatever MBCE and HBCE and their architecture (see Fig.3, HBCE guides MBCE and vice versa), and then designing hybrid-CE. Such approach is generally high because we can apply development processes several times. The development process begins with following top down approach, ending with a bottom-up approach. It's more flexible, but less reusable in an SDLC. Because many aspects which depend on the actual Xand their A involved in the P. From (3) and (4), we can deduce Hybrid-CE such as:

$$Hybrid\text{-}CE = f(X, A, P) \quad (5)$$

Representation of hybrid-CEs.

One key element of the CrowdCE model is a hybrid-CE, as Fig.4shows. Each hybrid-CE is a specific requirement for a specific MBCE and HBCE to be designed, developed or modified to meet development needs. The specifications for what MBCE is going to be needed: From hybrid-CE, we can represent all the MBCE and HBCE embedded in the various processes identified in the task. From the MBCE specifications, all of the requirements can be and appropriately distributed into three categories of IT technologies that link to the three-classic IT architectures

(data, applications, and technical architecture). From the HBCE specifications, all of the requirements can be and appropriately distributed into four categories (Verification and validation, interpretation and analysis, content creation, and content Access)[*15*], which play a vital role in understandingthe behavior of workers and their skill, and then the HBCE-design.For example, a process may require very different actions depending on whether a participant is a new or existing with a task-achievement history.From hybrid-CEs, we can build software specifications, which has all the HBCE required in the various processes identified in the task. The specifications for what the process is going to be needed.

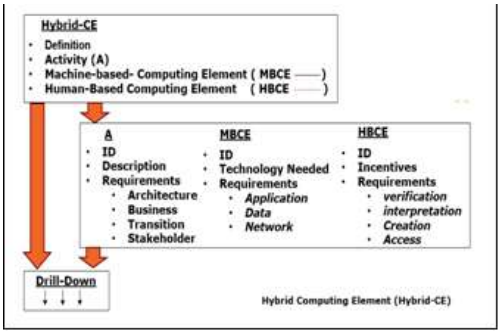

Figure 5. Representation of hybrid-CEs.

CROWDCE FOR SDLC

However, a novel blend of CEs requires a new model to let humans easily provide an HBCE and to efficiently deal with interactions through the SDLC. Current crowd computing models can't sustain hybridity because conventional CEs don't offer suitable workflow interfaces for developing applications, and current models don't generate interaction patterns to let human actors efficiently deal with requests. As indicated in the review of TopCoder's model [*16*], we believe that CE design and management is an essential requirement if you wish to empower the crowd. Fig.5 shows CrowdCEand the supported models that the CrowdCEdepends on to address these above requirements.

Decomposition and software development model

When designing an architecture for the CE, it is important to attempt to capture explicitly the envisioned variations that will occur between instances of it. Thus, CrowdCE is based on *The Architecture Based Design (ABD)* method [*8*] as it is shown in Fig.6; it works as recursive architectural drivers that have been determined with confidence. It is used for designing in parallel the high-level software architecture for a product line or long-lived applications. It fulfills functional, quality, and business needs at a level of abstraction that allows for the necessary development. Moreover, it provides a series of clear steps for designing the conceptual software architecture. In our context, one output of ABD model is a collection of hybrid-CEsthat constrain the execution of MBCE and HBCE. In detail, the inputs to the ABD method are a list of both abstract and concreterequirements.

A requirement is decomposed into items for which the crowd has the freedom to choose a solution. The decomposition is also worked from the point of view of the CE to determine where the MBCE or HBCE will be executed.For example, an MBCE is a CE that guides the decision a crowd must take. It is considered a conceptual interface that encapsulates the knowledge of data input and output. When reasoning about an architecture of CE, it's important to inspect it from an assortment of different perspectives. For example, a skill and complexity perspective that needs certain types of CE for developing solutions. According to the ABD method, CrowdCE begins the decomposition of each CE with a set of requirements (both functional requirements and quality attributes), and a set of constraints. However, we consider HBCE in the ABD method is worked well with coarse-grained variation at a granularity that has the effectiveness on the conceptual architecture, and MBCE works well with commonalities (fixed points within the HBCE).

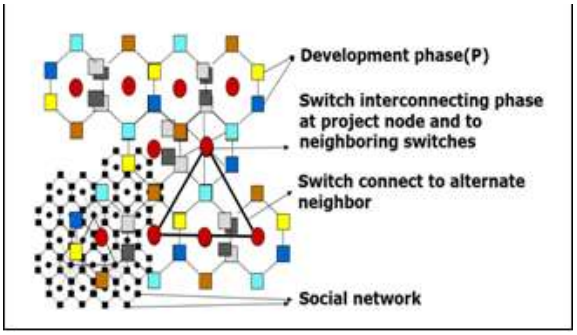

Figure 6. High-level CrowdCE for SDLC.

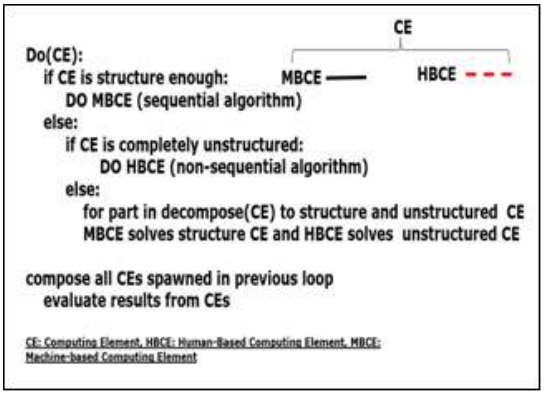

Figure 7. Decomposition and software development model (modified from [*8*]).

The pseudo code is presented in Fig. 7. We inspired execution model from a fork–join model of parallel computation [*9*], such that execution branches off in parallel at designated points (CEs). Parallel sections might HBCE or HBCE recursively until a certain CE granularity is reached, e.g., divide and conquer paradigm. Initially, a computation consists of a single CE and is assigned to some processor. An example, CrowdCE maintains several CEs of execution and schedules these into N processors whatever HBCE or MBCE. Each processor that has a current CE to execute.

Distribution and social computing model

Looking into the future, when the billion HBCEs are submitted, a concrete topology, management and representation of huge Hybrid-CE are needed. A Honeycomb structure (HCS) model as it is shown in Fig. 8; it is proposed to support CrowdCEmodelfor representing the Hybrid-CE and in two aspects 1) distributed computing resources, and 2) Social computing phenomenon. It will be briefly discussed each of them.

1. *Distributed computing resources*

The HCS supports the composition of distributed computing resources in the form of processor cores and field programmable logic blocks, distributed storage resources, programmable I/O, and all these resources are interconnected by a switching fabric, allowing any chip resource to communicate with any other chip resource[*10*].

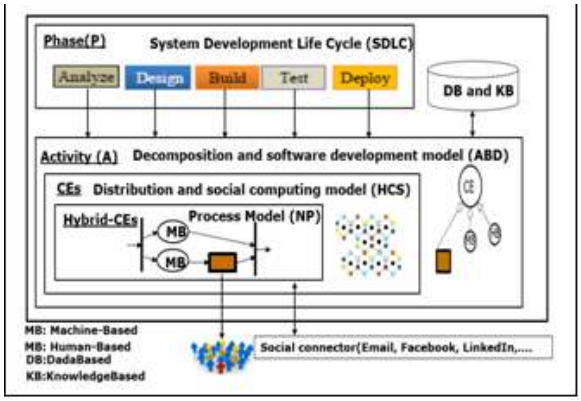

Figure 8. A pseudo code for decomposition and software development model.

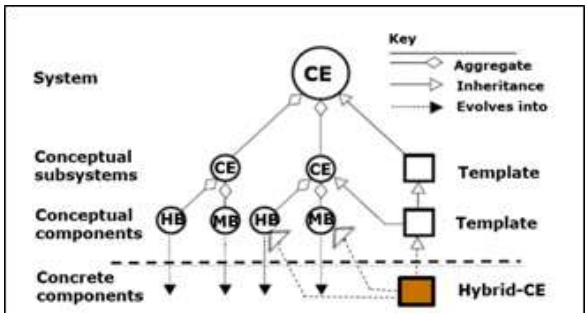

Figure 9. Distribution and social computing model.

As important as a good approach is a good representation which seeks to derive an executable functional system specification. Since the execution will consist of many concurrent Hybrid-CE and a very parallel architecture, this step is the first and perhaps the most important one towards the execution. This step basically determines the granularity and the atomic CE that eventually will be executed on specific Hybrid-CE. Moreover, in this step we determine the major communication requirements between both hybrid-CE and crowd. However, this step faces many problems such as optimization problems (e.g., what kind of basic HBCE is required).

2. *Social computing phenomenon*

Traditionally, crowd used the Internet to simply, create, modify, share, discuss, develop, design, and solve Internet content. This represents the social computing phenomenon, which can now significantly impact a user's reputation. HCS may support CrowdCE to manage social computing activities such as identity, conversations, sharing, presence, relationships, reputation, and groups. Here's a brief definition of each activity in our context.

- *Identity*: It is a way of uniquely identifying a member of the crowd
- *Presence*: It is a way of knowing who is online.
- *Relationships*: It is a way of describing how two members in the development phase are related (e.g., in Flickr, people can be contacts, friends of family)
- *Conversations*: It is a way of messaging to other members through the system
- *Groups*: It is a way of forming communities of specific work.
- *Reputation*: It is a way of knowing the status of another member of the system (who's a good member? Who can be trusted?)
- *Sharing*: It is a way of sharing things that are meaningful to participants,e.g., like photos or business model.

Process model

In Fig.9, a trivial CrowdCE process in Cilk-like syntax [*17*], which means that a computation can be viewed as a directed acyclic graph with a single initiator (start of CE) and a single decision (end of CE). Each node in this graph represents either an MBCE or an HBCE. MBCE produce the logical structure of parallelcomputations, HBCE produces unstructured logically parallel computations, and edges represent serial computation.However, the process model consists of four steps which are explained below.

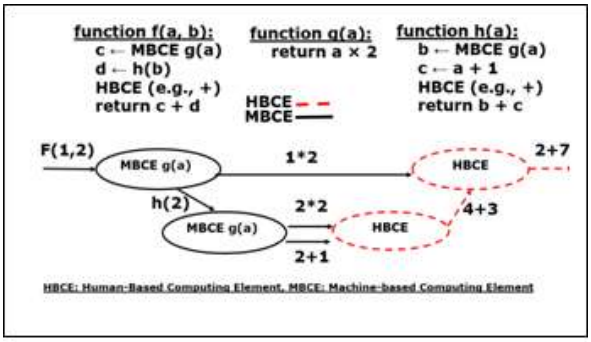

Figure 10. An example for process model in CrowdCE.

3. *STEP 1: Design as ABD.*

The process model begins to post the crowdsourcer's requirement model (e.g., post a manual sketch of thehigh-levelbusiness model). The model does not depend on any particular process in the analysis phase and thus, existing processes may be followed. Afterthe analysis phase, the design of the to-be-implemented application is made by technical architects (MBCE). The ABS can be visualized as a tree (e.g. Fig.6., above). Every hybrid-CE in ABD is a

tangible collection of HBCE and MBCE. They can be represented and realized through code. It clearly defines the role of HBCE and MBCE in one frame. Then, it will be linked together by communication linksfor constructing ABD tree. Since the ABD is a tree, the developer of one hybrid-CE does not need to aware where the parent and child template.

4. STEP 2: Specify user-experience HCS model

Once the ABD tree has been developed, hybrid-CE consists of reference test cases relating to the development process (P) and a textual description of the role of the different stakeholders(X). A switching module for communication and integration can be made without the knowledge of the way in which the actual development is done. Essentially, CrowdCE depends on the principle of the user experience honeycomb model [*18*][*19*] and other related works [*20*].Here, the development of an application can be made without depending on the predesign interfaces (i.e., without knowing the actual class or phase names which the crowd will develop later).

5. *STEP 3: Implementation*

Typically, every leaf of the ABD is an individual hybrid-CE is managed by HCS, with the completion of the hybrid-CE, HBCE is designed and posted to the crowds via a social network (e.g. Facebook).

6. *STEP 4: Validation and Verification*

As the crowd incrementally works with the hybrid-CE and submits the HBCE, unit tests which rely on HBCE and hidden tests which rely on MBCE are done. If these categories of test cases pass, the SDLC can be appropriately ended and deployed.

B. The collaboration Model

The role of the collaboration model is to translate the Hybrid-CE structure as described in Section III-E, which drives its actions. In a collaboration model, the MBCE's plan should always be governed by the HBCE, regardless of which CE is performing an activity at any given phase. At its core, the collaboration model is implemented as a state machine (Fig.10) commanding the SDLC throughout the collaboration, and triggering the appropriate social behaviors. As described in the previous steps, once a task is requested as a crowdsourcing, the MBCE is engaged to perform the hierarchical task jointly with an HBCE. When MBCE executes a hierarchical task, the role of another MBCE is to push the tasks onto a stack and perform each of the actions based on the preconditions of the HBCE.The model's states are defined, in flexible order of a typical collaboration, as follows:

- *Hybrid-CE-Next*:The initial state of the system, in which the MBCE evaluates the current social skill, and acts upon thisevaluation.
- *Ask For Do*: If MBCE is capable of achieving the next activity of the development phase, it will offer to take its turn, else next sentence.
- *Ask For Help*: MBCE will ask for help from the HBCE or itwill decompose the task into its constituent MBCE and HBCE and recursively pushes them to the stack.
- *Wait*: Waiting for a support from the other CE.
- *Execute*: If the MBCE is executing the current action, this happens in this state; if the HBCE is executing a step, the MBCE waits in this state.
- *Investigation*: Establishing common ground by looking through facts or evidence and come up with a decision.

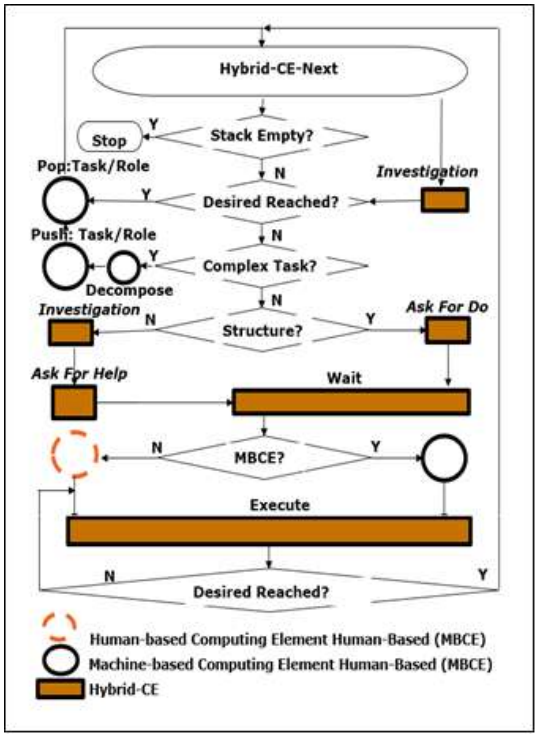

Figure 11. A schematic view of the task collaborator model. Note that the 'Wait' state can be terminated by both explicit and implicit turn taking on the HBCE.

II. EVALUATION

Our SDLC model in comparison with anothermodel is visualized in Fig.11.

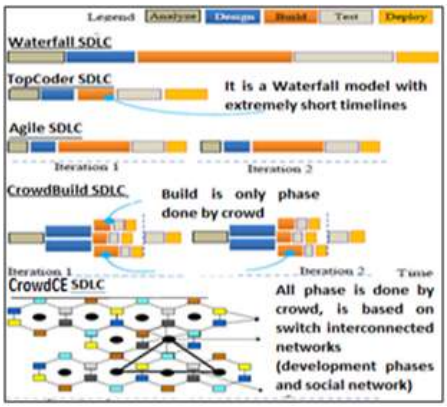

Figure 12. Comparison of different SDLC (modified from [*21*]).

In particular, the CE management framework is capable of creating hybrid-CE that can be developed in mesh style. We estimate a comparatively larger amount of effort in the analysis phase. (e.g., the decomposition and the representing of CE). However, the other phase of SDLS of our model is expected to be significantly small due to mesh execution unlike the other phase of SDLS in other models.

The CrowdCE lets people supply HBCE based on their skills and expertise. It acts as interaction interfaces toward humans, letting users develop various HBCE for different development activities indicating their ability and willingness.

C. Experimental study

To validate CrowdCE, we initiated an experiment of a software development application to build a multiple intelligencequiz "What does the scientific model say?" We chose to develop a scientific modelhelping the professor to generate multiple intelligence quizzes. The system will allow a member of the crowd (students) to sign-up for an account, post scientific model, and ask the other member to decompose this model into slices (shape and concepts). This is the first phase (posting the crowdsourcer requirements) and so on to the next phases of SDLC, as Fig. 12 shows.In each development phase, CrowdCE used four Hybrid-CEs (artifacts) to address design problems.

1. Where the students know, but they do not understand. Questions to complexity topics, learning values and emotions need to be asked directly to provide ‘emotional intelligence’ guidance for other students.So, they can anticipate this ambiguity and elicit sensitive tacit knowledge. The Common Ground question is to be more sensitive to the student’s background, feelings, and culture.
2. Where the students have some awareness of the necessary knowledge, a question for next task can be asked directly.
3. Moreover, most MBCE in the artifact involve reasoning and exploring the implications of the topic or domain boundary.

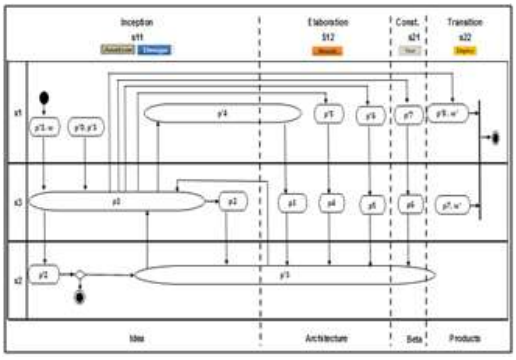

Figure 13. A collaboration model for ISSRsmartquize

4. Where the students know "What and How" scientific model is, that happens when they see the visual model. Most MBCE in the artifact involve prototypes, storyboards, and mock-ups; however, simulations and visual ontology may have further potential.
5. Where the student need to be integrated with social media, the collaborative creative quiz is empowered. Most MBCE in the artifact involve designing sociotechnical elicitation systems, e.g., e-communities, to develop new design ideas.

We chose this application as itscharacteristics are suitable for crowd work. Typically, it has a business logiclayer, knowledge base and a database. The system was called ISSRsmartquize.This work is based on a part of the student' course (ISSR-SWPM-2015).

In each phase, the authors of this paper assumed different roles as technical architects for the SDLC. The results captured through the experiment are in TABLE I.In the *analysis phase*, functional requirements were gathered, the general concepts built and the CE designed. At this phase, with the requirements clear, we decided on the technology to be used (PowerBuilder) and the infrastructure (SQL). We then used a simple interaction technique to have a baseline for the amount of business rules needed to build the application (e.g., Drag and Drop). In the *Design phase*, the ABD for the application was built. The system (ISSRsmartquize) has clearly decomposed CE, and generated a hybrid-CE as a user interface, based on predesigned MBCE. An MBCE housed common mechanisms. While an HBCE ties the MBCE with the business rules, MBCE forms common mechanisms using the database and knowledge base for forming the hybrid-CE to hold the data.. In the*implementation phase*, all the related HBCE and MBCE were made into a single hybrid-CE (DataWindow), while each DataWindow was an independent task. Thus, the ISSRsmartquize system comprised of 8 Hybrid-CE to be developed, 7 of which could be developed in the mesh (all PowerBuilder programming CE).

From 1, 2, 3, 4, 5, we can deduce developed work is,

Developed work (w') = w'(W, X, P, CrowdCE, S) (6)

Where,

- Work (W = w, w'), where w is a work before development, and w' is a work after development.
- Stakeholder(X= x_1,x_2,x_3), where x_1 is arequester, x_2 is a provider at the same time x_1 is a member of x_2, and x_3 is a social platform (marketplace).
- Activity (P=p, p'), where p is x_3-work (Machine-Based Computing Element (MBCE)), and p' is x_1-work or x_2-work (Human-Based Computing Element (HBCE)).
- CrowdCE = is a network of P between X to develop W (W is a collaborative software), as the following sequence:
  - p'0= Setting intensive, motivation and time constraints related to the specific activity (p'3).
  - p'1= Ask x_2 to sketch a specific work (w) for the specific domain model as they use.
  - p'2= If x_2 accept then p1, then p2 else reject.
  - p1= Manage registration and communication models for x_1 and x_2, and store x_2' work.

- p2= Generate a form to accept x_2' sketches then p'3.
- p'3= Sketch, design, test, and evaluate according to x_1-question and provide their model.
- p'4= Ask x_2 extract concepts from specific sketches.
- p3= Generate a form that has sketches associated with specific widgets, references (as knowledge-base) and x_1-question (task) with full task description.
- p'5= Ask x_2 to design (classify, set relationships or set dependency between concepts).
- p4= Generateconcepts form with the specific widget and related information to help x_2 and x_1 for doing their task (ontology building).
- p'6= Ask x_2 to design a work area referenced to the knowledge-base.
- p5= Verify and validate x_1 and x_2 work (loop deduction or violating a domain constraints).
- p'7= Ask x_2 to test the designed work which is referenced to the knowledge-base.
- p6= Generate designed work associated with the designed rules, constraints and data to be tested.
- p'8= Evaluation.
- p7= Evaluation

- SDLC (S=s1, s2), where s_1 is an engineering w, and s_2 is a production w'. s_1 is decomposed to s_{11}, and s_{12}, s_2 is decomposed to s_{21} and s_{22}, where
- s_{11} is an inception phase. It is to achieve concurrence among X on P, such as p'0, p'1, p'2, p'3, p1, and p2, and producing general idea about w.
- s_{12} is an elaboration phase. It is to build an executable architecture prototype for w, such as p'3, p'4, p'5, p'6, p3, and p4, and producing general Architecture for w.
- s_{21} is a construction phase. It is to manage, control, and process optimization for w, and testing against predefined references, such as p'7,and p6,and producing beta releases of w'.
- s_{22} is a transition phase. It is to build training plans and acceptance criteria in the requirements set, such as p'8, and p7, and producing products of w'.

D. Results from the Experiment

Based on results of the Experiment **(**Table 1**)**, all type of hybrid-CEs were executed, the work was very simple (e.g., the child can share). The business code quality was checked through expert.The experiment generated interesting results. Of the 140 CE posted, only 120 were successfully completed. The deadline for each hybrid-CE was 1 days. The results of the experiment support our primary claim of the CrowdCE as a feasible model for the SDLC through the Crowd. In particular, the CE management framework is capable of creating hybrid-CE that can be attempted in mesh style. The overall complexity of developing the system through our model is well comparable with that of traditional development. The results of the experiment are encouraging and there are multiple tracks for the further Crowd work.

III. RELATED WORK

With the continuing internationalization of the economy and contributions by active participants from virtually anywhere [*22*], the last few years have seen a sea change not only in technology and innovations in business models, but also in the development of new attitudes toward technology, business, and work. This change has to do with lowered barriers to entry into the profession. Seeking a new generation of software developers who are not formally trained as software specialists. Such as in [*23*], we weave visual analysis of social relationships into software development, leading to the notion of relationship-aware software [*24*], social machines [*25*] and social computing[*26*]. CrowdCE uses the metaphor of human social relationships,

TABLE I. RESULTS OF THE EXPERIMENT

Phase	ActivityDescription	No. ofsubmitted work	Level of complexity
Setup	Effort in set-up of tools needed to execute CrowdCEapproach		Simple (Routine matters)
Analyse	Preparing technical Requirements	6	Simple (Entry and learning)
	Preparing Use Cases		High (Specialist)
Design	Joint design computing element needed	12	Medium (Job class)
	Software template	4	Medium (Job class)
Build	Crowdsourcer business model	4	Medium (Job class)
	Messages exchanged	40	Medium (Job class)
	HBCE completed	150	Simple (Entry and learning)
	Reviewing the completed HBCE	40	Medium (Job class)
	Code written by the Crowd incompleted task	35	High (Specialist)
Test	Integration test cases tha passed	30	Medium (Job class)
Deploy	Deploying application	1	Medium (Job class)

but at the simplest and complex levels of human skills. CrowdCE takes into account other software which interacts with to establish a unifying abstraction model that is used for specifying relationships between HBCE and MBCE.

Due to space constraints, we present an overview of relevant classes of similar models, their typical representatives, and compare their principal features with the CrowdCE. Based on the way the workflow is abstracted and encoded the existing models, they can be categorized into three groups: a) programming-level model; b) parallelcomputing model; and c) process modeling. Like CrowdCE, there are developing level approaches focus on building a set of libraries and language constructs (such as if-then-else in CrowdCE) allowing a family group of developers to instantiate and manage CEs to be performed. Unlike CrowdCE, the existing models do not include the design of the CE decomposition model itself, and, therefore, have to predesign CE and then depend on commercial

platforms, such as CrowdDB [*21*] [*27*] andAutoMan [*28*].At the same time, CrowdCE like other parallel computing models, it depends on the divide and conquer methodology that decomposes complex CE into a set of MBCE or HBCE. Such as Turkomatic [*29*], Jabberwocky [*30*], Crowdlkang[*30*].

Skilled crowd work has been applied for an SDLS most notably by TopCoder [*16*]. To the best of our knowledge, there hasn't been any other specific model developed for the SDLC through the Crowd. A few inspections on the TopCoder platform can be made. First, the work deliverable (CE) in the challenges are not decomposed enough. So, it takes more time (for example, the entire CE of a particular development is done by a single member of the crowd). A high-level goal is not decomposed into atomic HBCE or MBCEthat can be integrated. Thus, there is no need to classify, combine and manage different skills of the crowd. TopCoder also follows the waterfall model [*7*] where each phase of the SDLC is done sequentially. The approach has an advantage such as it supports the re-use of components developed in earlier applications. However, it still rewards the efficient and smart crowd members (who can complete the technical CE a short duration). That means, it does not cover all levels of different skills or it does not empower people enough. Because we believe that the true value of the crowd can only be empowered as the participation improves and broad participation can be promoted through simple HBCEs that can be tackled by almost any member of the crowd.

IV. CONCLUSION

In this paper, we have presented CrowdCE as a model for the software development life cycle where the development phases can be done in a mesh style by the crowd. Using our model we have developed a smart application using crowd work. Our experiment showed the computingelements management framework is a feasible model to decompose a software application into atomic programming computingelements interms of human-based computing elements and machine-based computing elements and automatically validate and integrate the types of elements. This model paves the way for using collective intelligence to be brought into crowdsourced software development, creating a more responsive, crowd-centered process.

REFERENCES

[1] Hong-Linh Truong. (2014) http://www.slideshare.net/linhsolar/tuwasesummer-2014-engineering-humanbased-services-in-complex-processes.

[2] Luis von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford, "CAPTCHA: Using Hard AI Problems for Security," Advances in Cryptology—EUROCRYPT, vol. 2656, pp. 294-311, May 2003.

[3] Luis von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum, "reCAPTCHA: human-based character recognition via Web security measures," Science (New York, N.Y.), vol. 5895, no. 321, pp. 1465-1468, Sep 2008.

[4] Bruno Norberto da Silva and Ana Cristina Bicharra Garcia, "Ka-captcha: An opportunity for knowledge acquisition on the web," in Proceedings of The International Conference on Artificial Intelligence, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press, 2007, pp. 1323-1327.

[5] http://www.activevos.com/. [Online]. http://www.activevos.com/blog/bpel/bpel4people-and-ws-humantask-1-1-reach-public-review/2009/12/15/

[6] Tarek Ali, Eman S Nasr, and Mervat. Gheith, "Socially Intelligent Computing—A Survey of an Emerging Field for Empowering Crowd," in Proceedings of The 9th International Conference on INFOrmatics and Systems (INFOS), 2014.

[7] Klaas-Jan Stol and Brian Fitzgerald, "Two's company, three's a crowd: a case study of crowdsourcing software development," in Proceedings of the 36th International Conference on Software Engineering, ACM New York, NY, USA, 2014, pp. 187-198.

[8] Felix Bachmann, Len Bass, Gary Chastek, Patrick Donohoe, and Fabio Peruzzi, "The architecture based design method," CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, vol. No. CMU/SEI-00-TR-001, 2000.

[9] Dongarra Jack, "Algorithmic and software challenges when moving towards exascale," in Proceedings of the ATIP/A*CRC Workshop on Accelerator Technologies for High-Performance Computing: Does Asia Lead the Way?, 2012.

[10] Ahmed Hemani; Axel Jantsch; Shashi Kumar, Adam Postula; Johnny Öberg; Mikael Millberg; and Dan Lindqvist, "Network on chip: An architecture for billion transistor era," in Proceeding of the IEEE NorChip Conference, vol. 31, 2000.

[11] Jan H. Kietzmann, Kristopher Hermkens, Ian P. McCarthy, and Bruno S. Silvestre, "Social media? Get serious! Understanding the functional building blocks of social media," Business horizons, vol. 54, no. 3, pp. 241-251, May–June 2011.

[12] Avi Segal, Robert J. Simpson, and Ya'akov (Kobi) Gal, "Improving productivity in citizen science through controlled intervention," in Proceedings of the 24th International Conference on World Wide Web, 2015, pp. 331-337.

[13] Aniket Kittur; Jeffrey V. Nickerson; Michael S. Bernstein; Elizabeth M. Gerber; Aaron Shaw; John Zimmerman; Matthew Lease; and John J. Horton, "The Future of Crowd Work," in Proceedings of the Conference on Computer Supported Cooperative Work, New York, NY, USA, 2013, pp. 1301-1318.

[14] Klaas-Jan Stol and Brian Fitzgerald, "Researching crowdsourcing software development: perspectives and concerns," in Proceedings of The 1st International Workshop on CrowdSourcing in Software Engineering, ACM New York, NY, USA, 2014, pp. 7-10.

[15] Ujwal Gadiraju, Ricardo Kawase, and Stefan Dietze, "A taxonomy of microtasks on the web," in Proceedings of The 25th ACM conference on Hypertext and social media, ACM New York, NY, USA, 2014, pp. 218-223.

[16] https://www.topcoder.com.

[17] Niraj Shah, William Plishker, Kaushik Ravindran, and Kurt Keutzer, "Np-click: A productive software development approach for network processors," Micro, IEEE, vol. 24, no. 5, pp. 45-54, 2004.

[18] Paolo Montuschi, Andrea Sanna, Fabrizio Lamberti, and Gianluca Paravati, "Human-Computer Interaction: Present and Future Trends," Computing Now, vol. 7, no. 9, September 2014.

[19] Victoria Karaseva and Ahmed Seffah, "The human side of software as a service: building a tighter fit between human

experiences and SOA design practices," in Proceedings of The Eighth International Workshop on Cooperative and Human Aspects of Software Engineering, IEEE Press Piscataway, NJ, USA, 2015, pp. 105-108.

[20] Tarek Ali, Eman S. Nasr, and Mervat Gheith, "PipeGraph: A Visualized Interactive Sensitivity Analysis Method for Rule-based Systems," in Proceedings of The 2nd International Conference on Industry Academia Collaboration (IAC), Al Azhar Conference Center–Cairo, Egypt, 2015.

[21] Anurag Dwarakanath; Upendra Chintala; Shrikanth N. C.; Gurdeep Virdi; Alex Kass; Anitha Chandran; Shubhashis Sengupta; and Sanjoy Paul, "CrowdBuild: a methodology for enterprise software development using crowdsourcing," in Proceedings of The Second International Workshop on CrowdSourcing in Software Engineering, IEEE Press Piscataway, NJ, USA, 2015, pp. 8-14.

[22] Wei Li, Michael N. Huhns, Wei-Tek Tsai, and Wenjun Wu, Crowdsourcing: Cloud-Based Software Development.: Springer, 2015.

[23] Frank van Ham, Hans-Jörg Schulz, and Joan M. Dimicco, "Honeycomb: Visual analysis of large scale social networks," Human-Computer Interaction–INTERACT, pp. 429-442, 2009.

[24] Vanilson Arruda Burégio, Silvio Lemos Meira, Nelson Souto Rosa, and Vinicius Cardoso Garcia, "Moving towards" Relationship-Aware" Applications and Services: A Social Machine-Oriented Approach," in Proceedings of 17th IEEE International on Enterprise Distributed Object Computing Conference Workshops, 2013.

[25] Vanilson Burégio, Silvio Meira, and Nelson Rosa, "Social machines: a unified paradigm to describe social web-oriented systems," in Proceedings of the 22nd international conference on World Wide Web companion, 2013, pp. 886-889.

[26] Ognjen Scekic, Hong-Linh Truong, and Schahram Dustdar, "Incentives and rewarding in social computing," Communications of the ACM, vol. 56, no. 6, pp. 72-82, 2013.

[27] Michael J. Franklin; Donald Kossmann; Tim Kraska; Sukriti Ramesh; and Reynold Xin, "Crowddb: Answering queries with crowdsourcing," in Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, ACM New York, NY, USA, 2011, pp. 61–72.

[28] Daniel W. Barowy; Charlie Curtsinger; Emery D. Berger; and Andrew McGregor, "Automan: A platform for integrating human-based and digital computation," in Proceedings of the ACM international conference on Object oriented programming systems languages and applications, vol. 47, ACM New York, NY, USA, Oct 2012, pp. 639-654.

[29] Anand P. Kulkarni, Matthew Can, and Bjoern Hartmann, "Turkomatic: Automatic recursive task and workflow design for mechanical turk," Human Factors in Computing Systems, pp. 2053–2058, 2011.

[30] Salman Ahmad, Alexis Battle, Zahan Malkani, and Sepander Kamvar, "The jabberwocky programming environment for structured social computing," in Proceedings of the 24th annual ACM symposium on User interface software and technology, ACM New York, NY, USA, 2011, pp. 53–64.