DISTRIBUTED SOFTWARE DEVELOPMENT MODELLING AND CONTROL FRAMEWORK

Lei Wu[1], Sharon White[2], James Helm[3], Yi Feng[4]

[1,2,3]Software Engineering, University of Houston-Clear Lake, Houston, U.S.A
wul,whites,helm@uhcl.edu
[4]Computer Science, Algoma University, Sault Ste. Marie, Canada
feng@algomau.ca

ABSTRACT

With the rapid progress of internet technology, more and more software projects adopt e-development to facilitate the software development process in a world-wide context. However, distributed software development activity itself is a complex orchestration. It involves many people working together without the barrier of time and space difference. Therefore, how to efficiently monitor and control software e-development in a global perspective becomes an important issue for any internet-based software development project. In this paper, we present a novel approach to tackle this crucial issue by means of controlling e-development process, collaborative task progress and communication quality. Meanwhile, we also present our e-development supporting environment prototype: Caribou, to demonstrate the viability of our approach.

.

KEYWORDS

Distributed software e-development, task progress model, workflow control, quality monitoring

1. INTRODUCTION

During the past decade, more and more software systems are developed towards new technology platform [11][12][15]. Meanwhile, internet-based collaborative software development becomes one of the most significant practices adopted by many software engineering practitioners [6,8][16][13]. However, the large scale of collaboration over web lacks of sufficient supporting techniques to efficiently monitor and control web collaboration activities[14][18]. In this paper, we propose a novel approach to tackle this critical issue. We mainly focus on the modelling and controlling of web-based distributed software development activity. The ultimate goal of this research is to facilitate the co-operative work for web collaboration in e-development environment.

This paper is structured as follows: in section two, we introduce the distributed development supporting environment framework “Caribou” and its architecture. In the following two sections, we mainly present two major parts of Caribou, namely the e-Development communication control module and the task progress control module. In section 3, we illustrate our approach in monitoring and controlling the distributed collaboration and communication. In section 4, we discuss the technique we used to monitor and control distributed development collaboration task status. In section 5, we give the conclusion of our approach and discuss future work.

DOI : 10.5121/ijsea.2012.3501

2. DISTRIBUTED SUPPORTING ENVIRONMENT FRAMEWORK

We aim at providing a comprehensive distributed development supporting framework, called Caribou, to systematically manage software development process and coordinate team collaboration. Caribou also works as an integrated platform to accommodate various tools and techniques that are applied in the development tasks. The ultimate objective is to maximize the efficiency and the success possibility for any internet-based software e-development project.

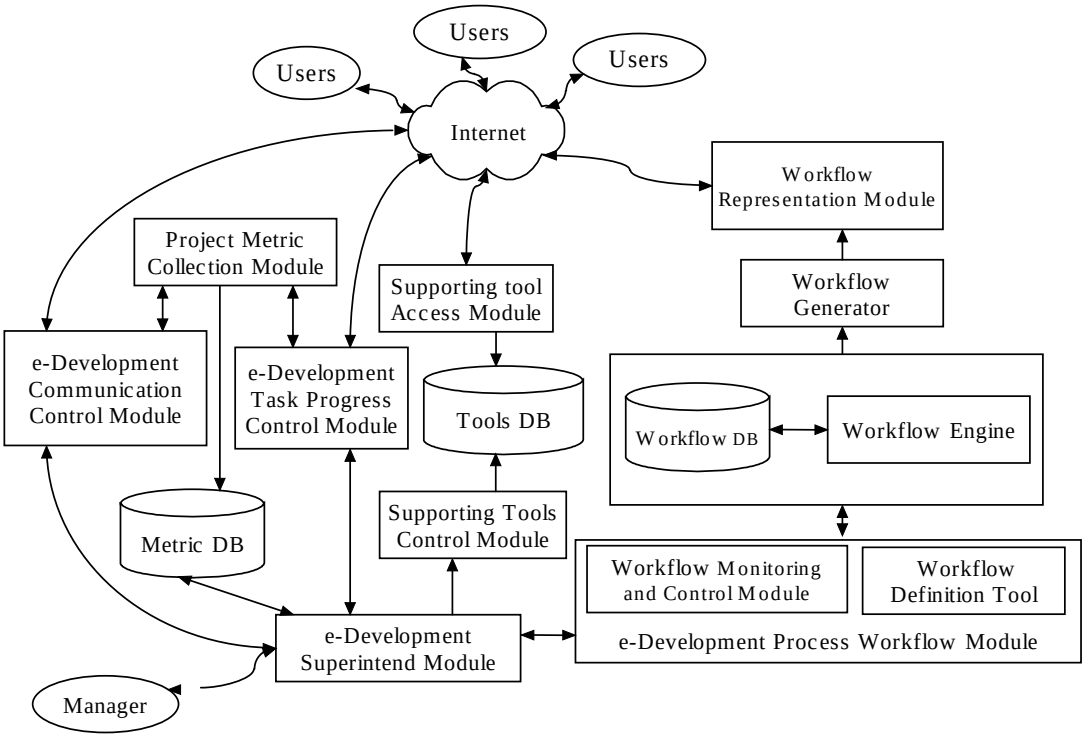


Figure 1: Caribou Architecture: Distributed Development Supporting Environment Framework

The Caribou prototype includes five major modules, see Caribou architecture, Figure 1. They are e-Development Communication Control Module, e-Development Task Progress Control Module, Supporting Tools Control Module, Project Metric Collection Module and e-Development Process Workflow Module.

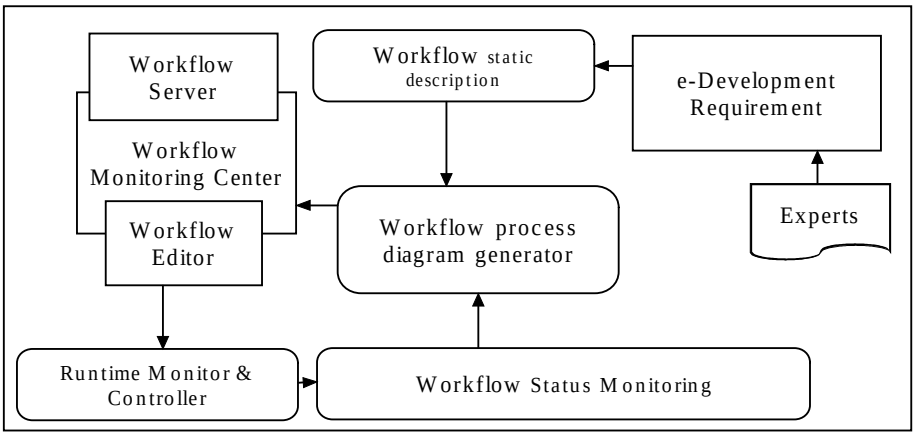


Figure 2: Caribou Workflow Generation Process

Figure 2 shows the Caribou workflow generation process. In the communication control module, a quantitative communication quality measurement mechanism is established to accommodate the need of e-development communication quality control. In this module, the performance of communication between developers will be enhanced through the control model and reinforcement model.

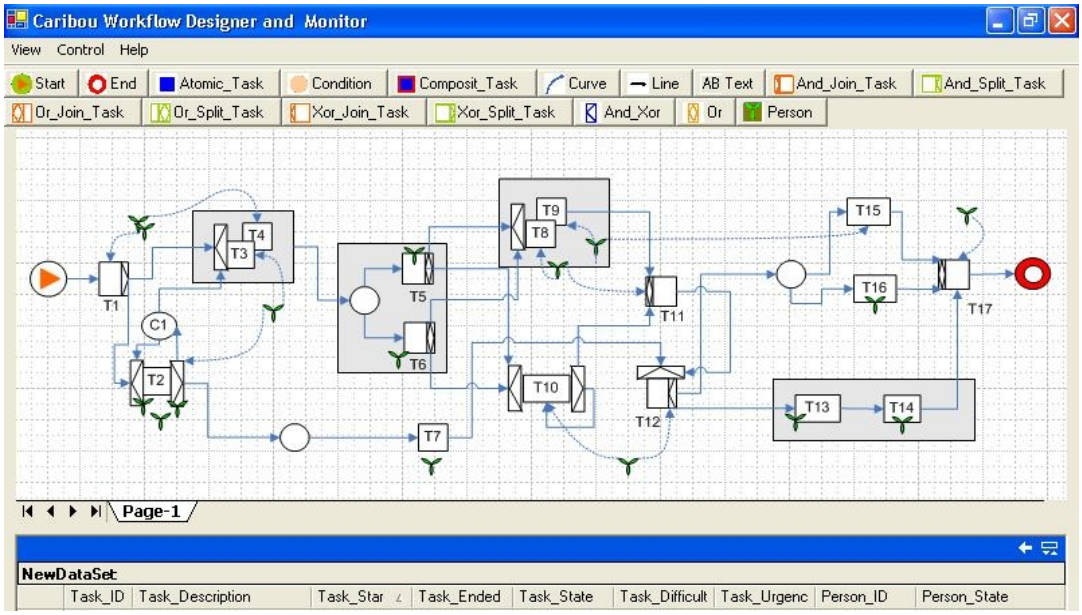


Figure 3: Snapshot of Caribou e-Development Process Workflow Modelling.

The major functionality of the task progress control module is to automate and monitor the e-development task progress, thus to provide an accurate measurement of project advancement. Supporting tools control module will host the related third-party tools and provide their services to the participants, thus to facilitate the detail e-development activities.

Project metric collection module will automatically collect and analyse a large range of process/project metrics, accommodate the quantitative control of the e-development project.

The challenges of developing e-Development Process Workflow Module in Caribou lie in the following aspects:

1. Design and implement a full fledged distributed workflow management system (WfMS). This embedded workflow server will later supply the automatic e-development process control. The feature of distribution means such a WfMS is internet enabled, thus maximize the flexibility of distributed application practice.
2. Model e-development activities using workflow definition formalism to specify their tasks and procedural constraints. Furthermore, it will support dynamical modification and adjustment of e-development process.

Figure 3 illustrates the e-development workflow modelling capability in Caribou.

3. COLLABORATIVE COMMUNICATION MODELLING AND CONTROL

In this section, we present the e-Development communication control module in Caribou. We first analyse and model human communication in an e-development environment. This model will later help us to monitor and control participants' communication activities within Caribou. Meanwhile, with the help of task progress control module, user will later be able to obtain a comprehensive view of the whole collaboration process by means of precise perceiving the development progress.

As we know, human is the primary factor in e-development. Collaboration among participants has two major forms: one is concrete working together to accomplish a task, the other is discussing with each other to solve some difficult problems. Many research works have shown that the communication between participants in a group is one of the most efficient methods of collaboration [8][9][18][22]. An important aspect of e-development communication is how individuals interact in a virtual group: web-based cooperation[19][20][23]. A cognitive-based analysis has been used to evaluate interaction effectiveness [3][1]. Further research emphasize on the coordination of communication [2][4][21][24]. Participants' cooperation efficiency largely affects the progress and quality of whole e-development project.

3.1. Networked Collaborative Software e-Development Communication Model

Meeting, discussion and pair programming etc. are various kinds of collaboration forms. The fundamental media that conveys those exchanges of information is communication. In a traditional environment, such activity is easy to obtain with oral language. While in a networked e-development environment, this type of information exchange is not that convenient to acquire. In most cases, lacking of an efficient way to monitor and measure communication activities remains one of the major obstacles of networked collaboration [5][10].

With the geographically separation and time difference, it's difficult to have a traditional meeting or discussion about a concerned issue. In many cases, people even may not be able to use on-line chat or video conference meeting to discuss, since their working hour may be reversed because of time zone difference. People would rather use email, instant messaging or bulletin board to exchange their opinions [7][21]. This type of communication raises another issue: how to quantitatively measure the quality of this sort of cooperation? How to reinforce the collaboration in terms of e-communication? The later one means, in e-development, if some people are sluggish

in providing the demanded information required by others, how can we adopt more efficient actions to largely avoid such situation?

Furthermore, if we have an urgent question and don't know who is responsible or is potentially able to answer it, the question becomes: who should we discuss with? Meanwhile, after we have published such urgent questions on e-bulletin board, and haven't gotten satisfied answer, what should we do next? If the proper person simply doesn't have time to browse discussion board, even though we know that there must be someone who has the answer, yet we still cannot trigger him/her out. This will eventually undermine the collaboration efforts. Therefore, how to effectively convey the concerned messages to proper people, how to secure the information solicitation mechanism and how to accurately measure the quality of collaboration by means of communication in a networked e-development environment are the three major goals for our research prototype.

To realize the ultimate objectives, we first build up our networked collaborative software e-development communication model, see figure 4.

There are two application scopes, namely communication scope and monitoring scope. The recording of a communication transaction and the evaluation of question/answer are performed in monitoring scope. This means, those processes are purely related with performance measurement purpose. While the communication activities are fully executed within communication scope.

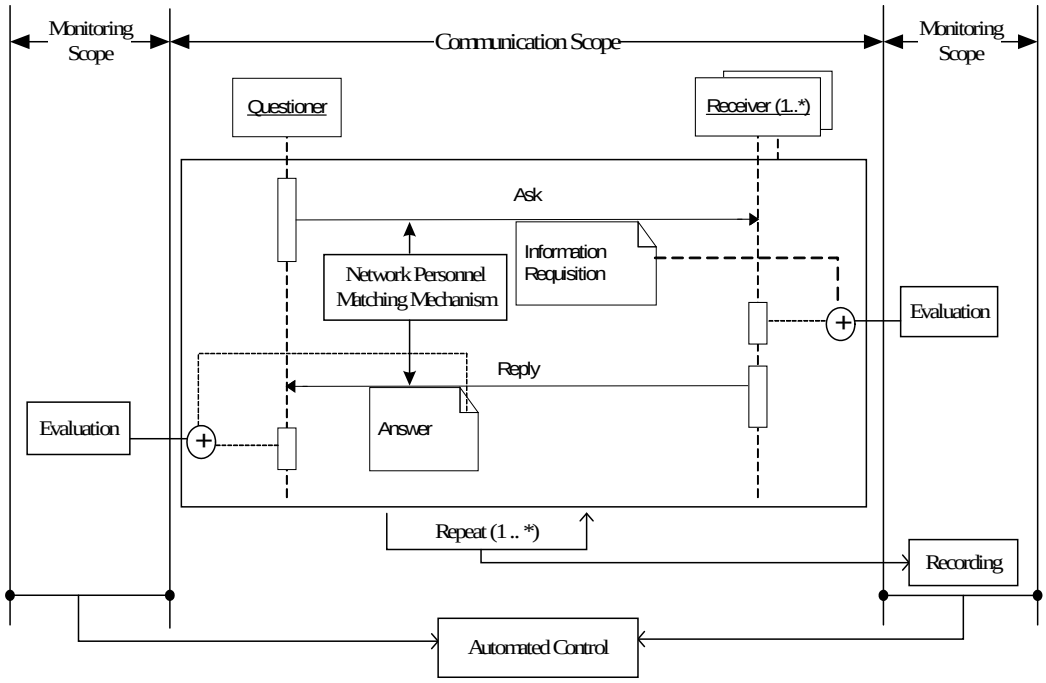


Figure 4: Networked Collaborative Software e-Development Communication Model.

We hereby apply this model to deal with four common forms of communication in a collaborative e-development environment. In such environment, the communication support program will help developers to easily interchange information and discussions [5,6]. In Caribou e-development supporting environment, this type of collaboration will be eventually computerized and recorded to evaluate the quality and progress of the whole project. Corresponding automated control will harmonize the performance with predefined standards. When questioner has a question or

solicitation of information, she may invoke a question. Based upon the network personnel matching mechanism (which will be elaborated in the following section), the question will be automatically conveyed to related person(s).

In a typical communication transaction scenario, questioner will fill in a predefined question head form to estimate some key metrics of the inquiry, such as urgency degree, importance degree and difficulty degree, etc. These sorts of information will be used to assist the automated control of the communication process.

A standard evaluation check form helps receiver to evaluate the question. The questioner also evaluates the reply by simply checking the quality tabular. The inquiry transaction may repeat for several times until the problem is solved or dead blocked. All these performance are recorded by e-Development communication monitoring module. The result will be processed by automated control module to trigger the corresponding control actions.

For example, a deadlocked question will be prompted to a higher-level group leader or technical coordinator to deal with; widely concerned questions may be presented to project manager, and request her to provide a general solution or suggestion; an extra delay of a question with high urgency degree will generate a caution message to group leader, etc.

3.2 Automation of Cooperative e-Development Communication in Caribou

To well observe human collaborative communication activities inside of software e-development projects, we have built up the networked collaboration communication model which has been presented in previous section. Now we apply this model to automate the monitoring and controlling measurements of personal communication activities in a distributed collaborative e-development environment.

e-Development Personnel Matching Mechanism: here we provide our solution to the first goal in our prototype, which is to effectively convey concerned messages to proper people. The solution for second goal will be presented in the following part of this section.

As we know, to automate the monitoring of cooperative communication activities, one important issue is to find the proper person to deliver the inquiry. There are mainly four communication forms defined in Caribou, in which a participant may involve. Based on these four forms, we've designed our network personnel matching methods to match the pair(s) of people to have a communication channel. This mechanism hereby implements the automated convection of questions to suitable person(s).

1.Direct Personnel: In this type of communication, the questioner knows who should be asked for. The matching mechanism will simply use the pointed stuff name or ID to directly deliver the questioner's requests to those who are expected to answer.

2.Direct Task-individual: In this communication form, the questioner doesn't know who should be asked for. Whereas he/she may know what tasks are related to the concerned information that he/she is inquiring.

The matching mechanism is to use the related tasks to trace the potential individuals who may have the answer. In a general case, if the task number is x, then all the persons who have participated in task-x may be considered as potential receivers. When there's more than one task that the questioner has marked as related to the concerned question, then the matching mechanism is to select those persons who have participated the most of the tasks. They will be most likely to

have a comprehensive view towards the question domain. The system divides people into several groups according to the number of tasks they have participated related to the question. Those who have participated the most tasks will be considered as the most likely possible receivers.

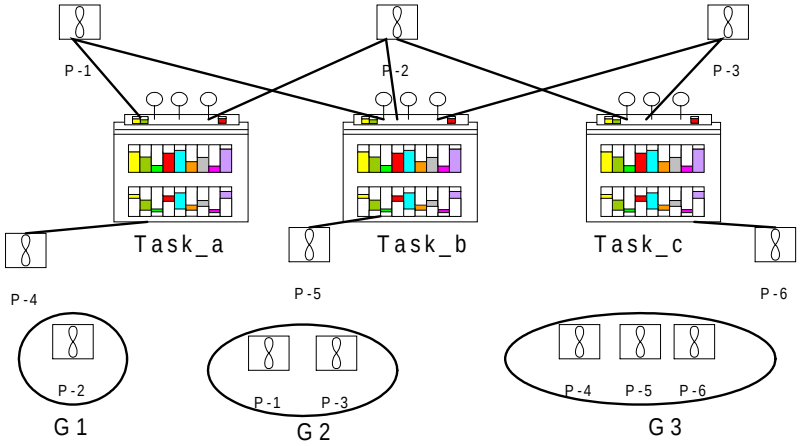


Figure 5: Direct Task-Individual Communication Model

As illustrated in figure 5, suppose there are three tasks marked as related jobs that concerned with the question. The matching process will first search the participants of these tasks in project database. Then, it generates several groups to hold personnel according to the task number they have participated in. Here we get three groups namely G1, G2 and G3. They hold personnel that have contributed to three, two and one tasks respectively. Finally, selection strategies may be applied to select potential receivers. As a result, the persons in G1 will be considered as the highest potential question receivers, while those in G3 will be considered to be the lowest potential receivers. The rational behind this strategy is that, if someone has more knowledge about the most of the tasks that the questioner is inquiring about, such person may be more suitable to provide pertinent answers.

3. Unknown Receptor: In some cases, the questioner may not be able to know who should be asked for, and even doesn't know which tasks are related with her question. The personnel matching mechanism will transfer this type of questions to e-bulletin board, group leader and technical coordinator etc.

4.Public Informing: When questioner just wants to provide some useful information/announcement for public, personnel matching module will convey it into public bulletin board.

The above four personnel matching mechanisms for distributed e-development will ensure an solution for each question that has raised during the development process, as illustrated in figure 6.

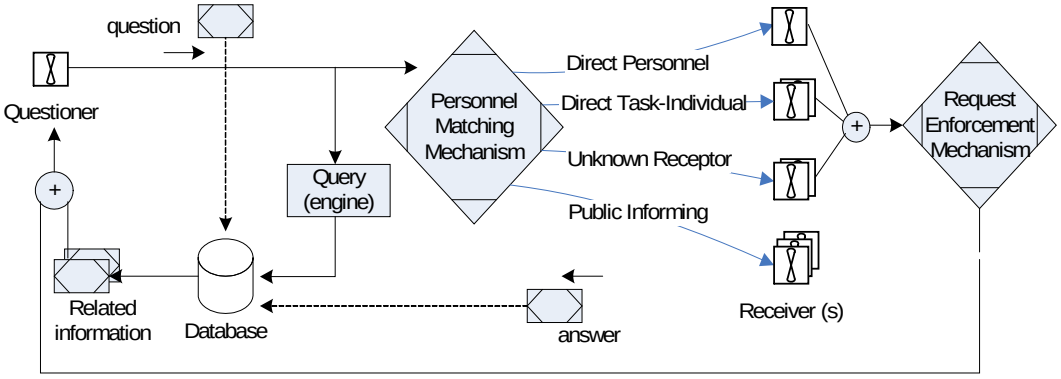


Figure 6. Caribou Personnel Matching Model: Question Transaction Process

When questioner (most left) publishes a question in a collaborative e-development environment, firstly, the question itself will be recorded in communication database; then, a search engine will query from this database to find out related information concerned with the question. The search criteria, for example, could be the same vocabularies that appeared in former question/answer pairs. Meanwhile, the personnel matching mechanism (center) will build up a channel to proper person(s). The request enforcement mechanism (most right) will ensure the elicitation of responses from those targeted receivers. The answer will also be recorded in communication database for other reference usage. The feedback from database and receivers will be presented to questioner.

The communication interaction may recur for several times, until a problem is solved or deadlocked, which will be triggered to the attention of the manager level personnel. The automated control module also deals with abnormal situations within Caribou communication module.

3.2 Request Enforcement: Performance Control

In this section, we'll realize our goal of securing the solicitation of desired information in e-development. That means, the answering of a question is no longer an option, or a spontaneous reaction. It is somehow a mandatory request in a typical distributed software development project. Moreover, such performance data will be recorded to quantify the evaluation of the collaborative e-development quality.

To fulfil this objective, we have established a request enforcement mechanism to help questioner squeeze out a high quality answer, as showed in Figure 7. In Caribou e-development supporting environment, there is a set of pre-defined dealing solutions. Each solution contains a collection of action scripts to invoke correspondent actions based on the pass-time length. It could be a remaindering to receiver after a short period of passed time; and if the time length is quite longer than expected response period, a reinforcement action will be taken.

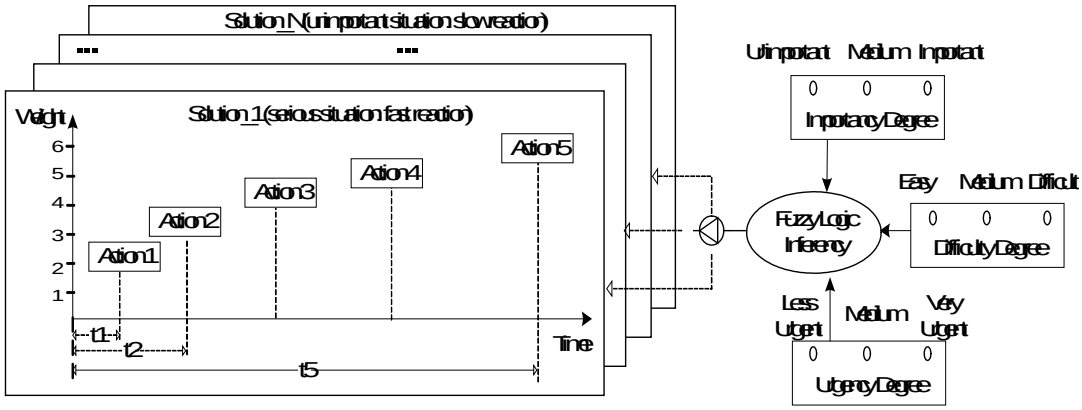


Figure7. Caribou Request Enforcement Model.

For each question, before it has been sent out, questioner has to mark estimated fuzzy values for three metrics, namely urgency degree (less urgent, medium, very urgent), importance degree (unimportant, medium, important) and difficulty degree (easy, medium, difficult). Then, based on these values, the Caribou communication control module will use fuzzy logic to figure out the suitable solution, and trigger the corresponded actions based on the waiting time intervals. These actions are labelled with weight value. Each weight represents the tolerance degree of the action when waiting interval exceeds a certain time-length of threshold.

The solution set (left) includes many solutions to deal with different types of conditions. For each solution, it includes several actions to cope with that condition based on waiting time intervals. Control module uses fuzzy logic and the fuzzy set of three metrics (right) to calculate the number of most suitable solution.

Therefore, with the help of Caribou personnel matching and communication request enforcement mechanisms, we are able to effectively convey the concerned messages to proper people, and secure the information solicitation. The dynamic performance data will late be used to quantitatively measure the quality of collaboration by means of communication in Caribou e-development supporting environment.

4. COLLABORATIVE DEVELOPMENT TASK MONITORING

In this section, we will present the e-Development Task Progress Control Module. Collaboration status monitoring and controlling is one of the most important issues in distributed development project. A dynamic supervision of on-going tasks has to be deployed. It will regulate both collaborative group members and external management, thus to ensure the development project to be in schedule and adopt proper actions in case of schedule slippage. In this section, we discuss our approach in fulfilling this objective. We apply autonomous task agent to facilitate the automation work. The goal is to strengthen the quantitative control of distributed development collaboration and reduce the inconsistency between different practitioners and organizations.

4.1. Web Collaboration

Like any kind of engineering practices, distributed software development is a progressive process. There is a time line to distinguish different stages of achievement. Meanwhile, this line also shows the progress curve of advance. It helps practitioners to track the historical performance as well as monitor and control the activities that are presently undergoing. Unfortunately, for a web-

based collaborative software e-development project, due to the distributive and dynamic nature of web collaboration environment, the time line is not easy to define. Web collaboration has made all the participants in a virtue development venue. All these may require practitioners to have an efficient way to monitor and control their widely dispersed development activities. We try to solve this complex problem by means of well defining the visual collaboration progress model. Furthermore, we apply autonomous agent to collect quantified data to measure the progress metrics in e-development project. In *Caribou*, we also have defined a set of control measurements corresponding to different progress deviations, and authorize agent to finally automate the actions to control development progress, thus to provide real-time support of e-development collaboration.

4.2. Collaboration Task Progress Modelling

In a collaborative software development project, the collaboration is consisted of many tasks that involve different individuals. To monitor the whole collaboration, we should have a clear view of each task's progress status. Task generally can be viewed as an elemental work unit as a conceptually whole that is performed by one or several persons. It normally includes concrete detail functions, performance steps and final objective. The major difference between task and function lies in task is a logical definition and participant number can be more than one; while function is a detailed minimum job that can be accomplished by a single person. The challenge lies in that, at any given moment, how can we measure the advance of task's progress? Our approach is to utilize agent to dynamically collect task attribute information, use predefined progress model to determine the progress metrics, and visualize them to reflect each task's progress status. To illustrate our approach, we first analyse task attributes. A collaborative e-development task has following two types of attributes: static and dynamic attributes.

Static attributes: includes task name/id_code, type, scheduled time, participant (number, id/names), anticipated output functions (name/code, task result description, number), steps (name/ id_code, description, number), scheduled resource (name/ id_code, type, quantity), ultimate objective (description), etc.

Dynamic attributes: includes consumed time, consumed resources (name/ id_code, type, quantity), undergoing development function (name/ id_code, number), importance (quality requirement), urgency (development time requirement), stable output function (name/ id_code, number), unstable output function (name/ id_code, number), undergoing step (name/ id_code), emergent event (name/ id_code, type, number), extra resource request (name/ id_code, type, quantity), distance from final objective estimation (heuristic measurement), health index (heuristic measurement), etc. A task's dynamic attributes strongly reflect its progress character.

We notice that, under certain conditions, some static attributes can also be changed into dynamic attributes. For example, at a given time, an emergency event occurs. The static attributes of participant number may be changed into dynamic attributes. A visual model, shown in figure 8, illustrates the collaborative e-development task progress model in Caribou.

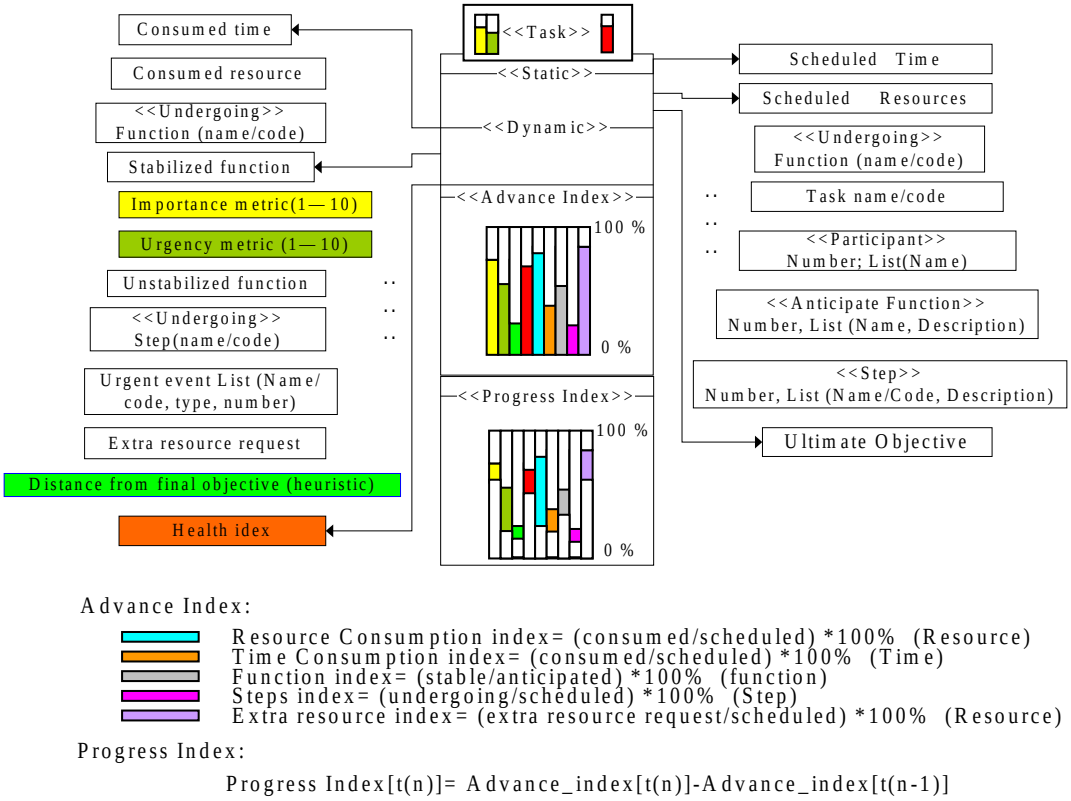


Figure 8. Caribou Collaboration Task Progress Model.

It visually represents the progress degree of each task inside of the whole collaborative e-development process.

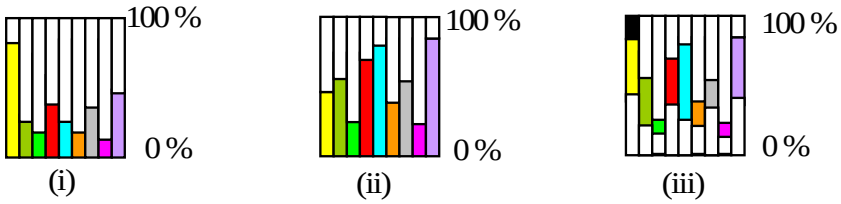
At any given time, it shows the static (right) and dynamic (left) attributes of the observation task (central). In the meantime, it shows the dynamic aspects in terms of advance and progress rate. The advance histogram (central) indicates the advance of this task; and the progress histogram (bottom) shows the abstract progress during a period of time. On the top beside <<task>> stereotype, there are three most important metrics, namely the importance, urgency and health index of the task.

Advance index: indicates the advancement of a given task at any time. It is measured by a group of metrics. They together reflect the advance degree of the task. Here we give the definition of them:

Resource_Consumption_Index= (consumed/scheduled)	(Resource)
Time_Consumption_Index= (consumed/scheduled)	(Time)
Function_Index= (stable/anticipated)	(Function)
Function_Quality_Index = (1 - instable/anticipated)	(Function)
Steps Index= (undergoing/scheduled)	(Step)
Extra_Resource_Index=(extra resource request/scheduled)	(Resource)

Progress Index: The activity progress of a given task will be represented by visualizing the difference of "advance index" between the start and end states during a period of time, see figure 9. They work together as a sign of progress to show the total activity increment results. Assume the time interval between two given moment is $\Delta t=t2-t1$ where t1=start time, t2=end time. To measure a certain task, or a group of tasks, we assume the period of observation time Δt is same. Therefore, we define progress index for each metric as:

$$Progress\_Index(t2)=Advance\_Index(t2)– Advance\_Index(t1)$$



(i) Advance Index at moment t = t(n-1)
(ii) Advance Index at moment t = t(n)
(iii) Progress Index[t(n)]= Advance_index[t(n)]-Advance_index[t(n-1)] (at moment t=t(n))

Figure 9. Calculation of Progress Index

At given moment time= t(n), when progress index value is negative, there will be a black column above the absolute value to indicate the negative increase, such as progress index of "Importance Metric" showed in our case (right graph (iii), the most left column, showed as yellow). Because at moment t=t(n), the importance metric values is smaller than that of one moment ago (time=t(n-1)). Therefore the progress index value is negative, which shows the decrease of progression.

Health Index: shows in general, the quantified health degree of a task. We use more sophisticated method to get the value. To avoid bias, we use both advance and progress index metrics to compute the assessment of more subjective metrics. Furthermore, these metrics can also be estimated according to cooperative organization's policy strategy. We define "Health Index" metric as:

$$Health\ Index= \Sigma\ (Pi*WPi)\ + \Sigma\ (Aj*WAj)$$

{i=1..N, j=1..M|N=the number of progress index; M= the number of advance index}. P is progress index and A is advance index. WP is the weight for progress index; WA is the weight of advance index. The weight represents the importance of each aspect from the organization's point of view.

4.3. Real-time Visual Collaboration Network

After building up models for collaborative e-development task, our next objective is to monitor the current status of distributed software e-development activities in Caribou. One crucial step to realize automatic monitoring is real-time collaboration activity data collection. Caribou e-development support system works as a general platform to serve collaboration among institutions.

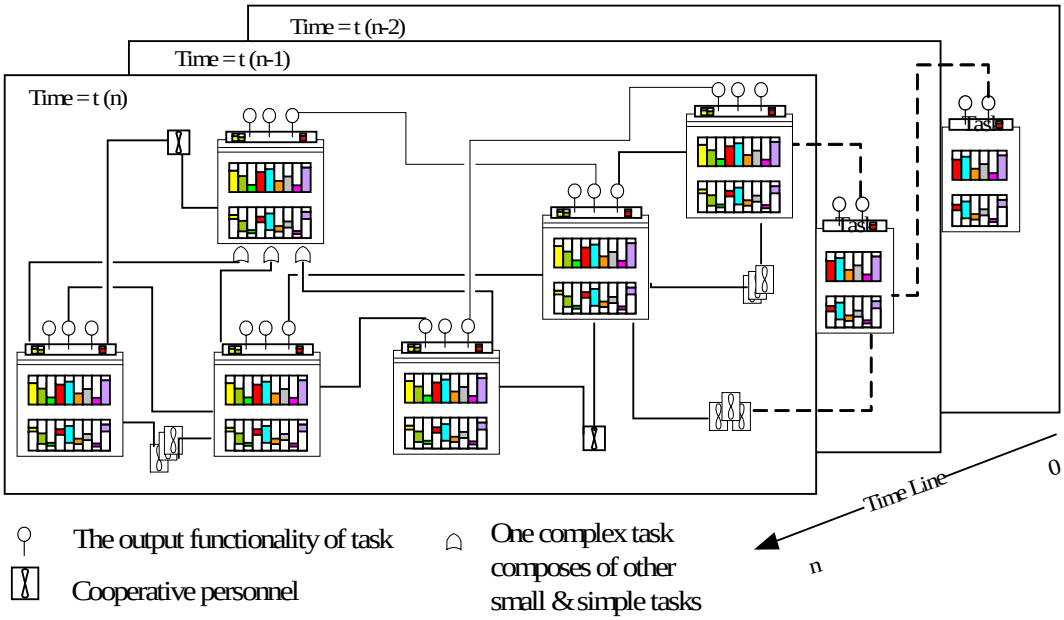


Figure 10. Collaboration Network

Meanwhile, individual participants use Caribou framework system as a way to either access project resources or collaborate with others without considering the time and space variance, see figure 10. Their performance is monitored and coordinated by Caribou environment. Task agent is used to automatically collect dynamic task attribute data and event information.

5. DISCUSSION AND FUTURE WORK

In Caribou prototype, we use graphic diagram to visualize the status of collaborative tasks, their relations and the whole project progress historic performance for an e-development project. In distributed collaboration development, task network has two dynamic aspects. One is the continuously generation and expiration of tasks; the other one is the constantly changes inside of existing tasks. At a given time, the whole development may have a certain number and types of tasks that are undergoing. After a period of time, some tasks may have already been finished, while some new ones have been generated to suit for current project progress need. Therefore, our whole collaboration network will regularly change its shape and organization. Each task has its own life-circle and may have many people cooperate within it; various events can affect its progress route. Caribou environment uses the dynamically collected activity data by to generate the visual web collaboration network diagram to facilitate the monitoring and controlling of an active e-development project. Each task also has its own output functionalities, namely the results it generates. For example, a task named “Implementing Class C” may have two functionalities: one is the usable class, and the other one is its usage documents. Such functionalities are the produced results. A task may use the output functionality produced by others. We use line to connect the request task and the output functionality of service task (represented by tiny circle). Two tasks may “cooperate” with each other by providing services to both. For example, task “Testing Component A” and “Testing Component B” may mutually use the functionalities provided by each other. When a task is finished, all the expected functionalities should be produced. To use the result provided by a historical task that does not lie on the same layer of the active tasks, a dotted line will be used to connect two tasks between two layers. The same will also be applied on participant personnel. Collaboration task network visually depicts the dynamic organization and status of tasks/personnel. Several people may work together on a same task; and one person can also participate in more than one tasks. The lines between tasks thus represent the

conceptual supporting relationships among them. Based on the monitoring data that has been obtained by task agent, we are able to simulate the past and present course of web-based collaborative software development process simultaneously. Furthermore, we may later be able to use these two types (present and past) of results to analyse the collaborative development performance.

Web-based collaborative software e-development involves many people working together without the barrier of time and space difference. However, the large scale of collaboration in a typical e-development project lacks of sufficient supporting techniques to efficiently monitor and control the distributed collaboration activities. In this paper, we have presented our novel approach and the prototype of Caribou: a supporting environment for distributed collaborative development, to tackle this important issue. In addition, we also provide solutions to automate the dynamic control of cooperation, thus to fulfil the goal of optimizing the e-development performance. In the following future work, we aim to build up service models for these collaboration monitoring and control service, thus to provide a general purpose collaboration assistant service over internet.

REFERENCES

[1] Zhang, J. (1998), "A Distributed Representation Approach to Group Problem Solving", Journal of the American Society for Information Science, Volume 49, Number 9, pp. 801-809.

[2] Kies, J., Williges, R., Rosson, M., (1998), "Coordinating Computer-Supported Cooperative Work: A Review of the Research Issues and Strategies", Journal of the American Society for Information Science, Volume 49, Number 9, pp. 776-791.

[3] Marsic, I. (2001) "An architecture for heterogeneous groupware applications". Proc. 23rd IEEE/ACM International Conf. on Software Engineering

[4] Preguiça, N., Legatheaux Martins, J., Domingos, H., and Duarte, S. (2000), "Data management support for asynchronous groupware". Proc. ACM Conference on Computer-Supported Cooperative Work (CSCW'00), Philadelphia, PA, pp.69-78

[5] J. Peng and K. H. Law, (2002) "A Prototype Software Framework for Internet-Enabled Collaborative Development of a Structural Analysis Program", Engineering with Computers, Springer-Verlag, 18: pp. 38–49

[6] Jon Derome and Kosin Huang, (2003), "Creating and Delivering Value with Collaborative Software development", Business Applications & Commerce , September 2003

[7] Martha L. Hause and Mark R. Woodroffe, (2001), "Team Performance Factors in Distributed Collaborative Software Development", 13th Workshop of the Psychology of Programming Interest Group

[8] Teasley, S., Covi, L., Krishnan, M.S., Olson, J.S. (2000), "How does radical collocation help a team succeed?" Computeer Human Interface Proceedings

[9] Nardi, B., Whittaker, S., and Bradner, E. (2000): "Interaction and Outeraction: Instant Messaging in Action", Proceedings of CSCW 2000, Philadelphia PA, December ACM Press, 2000, pp79-88

[10] M. de Jonge, E. Visser, J.M.W. Visser, (2001), "Collaborative software development", Software Engineering (SEN) SEN-R0113

[11] Szabolcs Feczak, Liaquat Hossain. (2011), "Exploring computer supported collaborative coordination through social networks", The Journal of High Technology Management Research, Volume 22, Issue 2, pp 121–140

[12] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb (2012). "Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository". In Proceedings of Computer-Supported Cooperative Work

[13] T. Wolf, A. Schröter, D. Damian, L. D. Panjer, and T. H. D.Nguyen. (2009). "Mining Task-Based Social Networks to Explore Collaboration in Software Teams". IEEE Software, 26:58–66

[14] McIntosh, S., Adams, B., Nguyen, T.H.D., Kamei, Y., Hassan, A.E. (2011), "An empirical study of build maintenance effort". In 33rd International Conference on Software Engineering pp141--150.

[15] I. Mistrik, A. van der Hoek, J. Grundy, and J. Whitehead ,(2010) "Collaborative Software Engineering". Springer

[16] D. Damian, I. Kwan, and S. Marczak. ,(2010) "Requirements-Driven Collaboration: Leveraging the Invisible Relationships between Requirements and People". In Collaborative Software Engineering, pages 57–76. Springer Berlin, Heidelberg
[17] J. Eckstein, (2010), "Agile Software Development with Distributed Teams: Staying Agile in a Global World", Dorset House, New York ISBN 978-0-932633-71-2
[18] D. Damian, L. Izquierdo, J. Singer, and I. Kwan. (2007). "Awareness in the Wild: Why Communication Breakdowns Occur". In Proceedings of The International Conference on Global Software Engineering (ICGSE '07), pages 81–90, Washington, DC, USA, IEEE Computer Society.
[19] D. Redmiles, A. van der Hoek, B. Al-Ani, T. Hildenbrand, S. Quirk, A. Sarma, R. S. S. Filho, C. de Souza, and E. Trainer,(2007), "Continuous coordination: A new paradigm to support globally distributed software development projects," WIRTSCHAFTSINFORMATIK, vol. 49, no. Sonderheft, pp. S28–S38
[20] F. Rodriguez, M. Geisser, K. Berkling, and T. Hildenbrand, (2007), "Evaluating collaboration platforms for offshore software development scenarios," in Proceedings of the 1st International Conference on Software Engineering Approaches For Offshore and Outsourced Development (SEAFOOD'07). pp. 96–108
[21] Deshpande, A. and Riehle, D, (2008),"The total growth of open source". Open Source Development, Communities and Quality (2008), 197--209.
[22] S. Bugde, N. Nagappan, S. Rajamani, and G. Ramalingam. (2008), "Global software servicing: Observational experiences at Microsoft". In IEEE International Conference on Global Software Engineering
[23] T. Nguyen, T. Wolf, and D. Damian. (2008), "Global software development and delay: Does distance still matter?" In Proceedings of the International Conference on Global Software Engineering,
[24] L. D. Panjer, D. Damian, and M.-A. Storey.(2008), "Cooperation and coordination concerns in a distributed software development project". In Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering, pages 77–80. ACM.

Authors

Lei Wu, Assistant Professor of software engineering at University of Houston-Clear Lake, Houston, U.S.A. His major research interests include software engineering with artificial intelligence, secure service-oriented architectures, software for robotics and embedded system intelligence, game software development, and pervasive computing. He can be reached at wul@uhcl.edu

Sharon White, Associate Professor of software engineering at University of Houston-Clear Lake, Houston, U.S.A. Her research interests include domain specification languages, architecture design languages, and software architecture. She can be reached at whites@uhcl.edu

James Helm, Associate Professor of software engineering at University of Houston-Clear Lake, Houston, U.S.A. His research interests include systems and software engineering, operations research, computer science, mathematics, physics, simulation, and modeling. He can be reached at helm@uhcl.edu

Yi Feng, Associate Professor of computer science, Algoma University, Sault Ste. Marie, Canada. Her major research interests include formal verification, software engineering, signal processing, and system description languages. She can be reached at feng@algomau.ca