

ZFlow: Workflow for Cooperative Editing System

Fei Zhu

School of Computer Science and Technology, Soochow University, Suzhou, China
Email: zhufei@suda.edu.cn

Xiaoxu Zhu, Qiang Lv, Yiyong Shi, Yang Yang and Qiaoming Zhu
School of Computer Science and Technology, Soochow University, Suzhou, China
Email: {xiaoxzhu, qiang, yyshi, yyang, qmzhu}@suda.edu.cn

Abstract—Enforcement of global cooperation urges the rapid growth of cooperative work. As a typical application of it, cooperative editing is widely used in many fields gradually, especially in knowledge intensive application. Workflow is a kind of automation technology by predefined process rules and tasks. It can bring us clarity in process control and management, which are two important facets in cooperative editing. Marriage of workflow and cooperation shows us an approach to achieve automation and convenient management as well as collaboration, thus gaining efficiency and regularity. We summarize the editing process flow as iterative phases including initializing, editing, coediting, reviewing and finalizing, which lays the foundation of incorporating workflow with cooperative editing. An activity-role based access control model, which uses activity to represent the editing task and uses role to effectively manage participants, is designed and applied in the workflow system. To get fine control grain, section, a logic part of the document, is used. We specially introduce the concept of subflow for section processing control and use section for cooperative editing control. To represent the workflow, we propose a workflow description language, which can be understood and processed by the workflow engine. As control logic in cooperative editing is quite different from that in individual editing, we put forward a novel workflow control algorithm, taking into account characteristics of cooperative editing. A temporal cooperative editing system applied the workflow system, showing it can deal with cooperative editing well, spare much human work, and offer clear process logic.

Index Terms—workflow, cooperative editing, activity, transition, flow control

I. INTRODUCTION

With the development of network as well as the popularity of distributed system, computer applications are now gradually transferring from single user mode to cooperative working by multiple users.

There are some special requirements in cooperative

editing by multiple users, which do not exist in individual editing environment, such as Microsoft Word. The first is real time responding. The operation demands by local host user are instantly answered by the machine. As for remote users, potential delay should be as short as possible. The second is that user group should be built on robust network. Effective and efficient control and management are necessary. The third one is to provide operation free editing which allows users to edit the document as they wish, thus requiring multiple concurrence management and maintenance.

More and more people make use of electronic digital equipment and appliance, such as computers, which are generally connected to network, i.e. the Internet, to edit documents. Such network connected electronic digital devices provide prerequisites for cooperative working. At the same time, the number of digital documents to be edited has been explosively growing for years, which would be counted by billion now. Furthermore, documents of the knowledge intensive era are much more sophisticated both in content and in structure than before. As a result, individual editing gradually shows its inability in dealing with these huge numbers of knowledge intensive documents. It is no longer fit for practical editing demands of the time being. Computer Support Cooperative Editing (CSCE), which is a typical application of Computer Support Cooperative Work (CSCW), emerges under such circumstance. It is welcomed by users from different aspects and develops rapidly.

As we know, procedure of editing a document is usually composed by six main steps: making a new document, editing the document, submitting the document for reviewing, reviewing the document, revising the document and finalizing the document. Definitely it has the characteristics of workflow, and therefore can be controlled by workflow. With workflow, documents, information or tasks are passed from one participant to another for action according to a set of procedural rules, which takes automation into editing, thus bringing convenience and regularity to human work. CSCE and workflow are two main mechanisms in

Manuscript received February 28, 2009; revised March 30, 2009; accepted April 15, 2009.

information processing. CSCE focuses on collaboration while workflow emphasizes on control and automation. Combining these two together will gain all benefit from workflow as achieving cooperative editing.

In this paper we propose a novel flexible workflow, called ZFlow, for cooperative editing system. With ZFlow, multiple users can easily coedit a document simultaneously, improving efficiency to a considerate level.

II. RELATED WORK

A. Computer Supported Cooperative Editing

Cooperative editing is process of multiple users fulfilling editing. It has a solid foundation during its forming and evolution. In present information society, human life and work styles are of colony, along with the characteristics of interaction, distribution and cooperation. At the same time, novel computing technologies including parallel computing and distributed computing, multimedia technology, database, and communication and network form fundamentals for CSCE. Moreover, concurrent engineering which is a methodology about integration, as well as parallel work which focuses on team work, pushes CSCE forward. CSCE is proposed under cooperative working and is based on harmony of computer technology, communication, and human work.

CSCE is an integrated application, including groupware which is a kind of software that supports cooperative editing, model for cooperative editing, communication among computers, human computer interaction, group decision support, coordinating system, distributed system, cooperative activity, organization, and cooperative supported artificial intelligence.

CSCE aims at cooperative editing by multiple users with computer support. It hammers at using computer and communication technologies to improve ability to solve problems more conveniently, more quickly, more flexibly and more comprehensively. Many practical applications take advantage of CSCE, such as Internet meeting room, remote meeting, email, desktop meeting system, and multi-editing system.

There is some typical applications in the area, such as CoWord and CoPowerPoint, by which the concurrence control model for CSCW editing has been well established[1-4].

CoWord[1] converts Microsoft Word into a real-time multiuser collaborative word processor. With CoWord, users can edit the same Microsoft Word document simultaneously over the Internet. CoPowerPoint[2] is a real-time collaborative multimedia slides creation and presentation system, allowing multiple users to create and present the same Microsoft PowerPoint documents at the same time over the Internet. CoPowerPoint supports unconstrained slides creation and presentation style, giving users complete freedom in their way of using CoPowerPoint to support individual and group work. Both CoWord and CoPowerPoint are powered by Generic Collaboration Engine which integrates a comprehensive collection of state-of-the-art collaborative technologies.

However, these two applications are not equipped with any workflow mechanism.

B. Workflow

According to WFMC[5], workflow is a kind of automatic execution processing, which completes transferring and execution within different executors by a serial of process rules, documents, information, or tasks.

Productions of workflow systems, such as Staffware, MQ Series Workflow, and InConcert, are process centric and support activities. The workflow module Webflow of SAP R/3 is also process-aware. Widely used ERP systems can be viewed as a set of workflow applications that are generally built on database.

C. Combination Cooperative Work with Workflow

Cooperation has become important especially in knowledge intensive applications. The marriage of cooperative work and workflow has showed some advantages. Toan Nguyen was devoted to the distributed workflows as a means to support large simulation applications[6], such as aircraft in flight dynamics. Nguyen's work focused on cooperative design workflow on wide-area networks. The use of distributed and composite workflows resulted in effective and seamless large-scale simulation.

Much work has been done in combination cooperative work with workflow. Some researchers took advantage of ontology to fulfill cooperation in workflow. I. T. Hawryszkiewicz proposed a metamodel for computer support systems for collaborative work on the Web[7], whose model provided ontology to support collaborative process modeling. Zhilin Yao introduced a workflow centric collaboration system based on ontology[8]. Yao made use of ontology to represent most collaboration elements and rules of the system, therefore enabling the system to represent collaboration knowledge. But Yao's system can merely interact with other systems and applications that should also be ontology based, limiting the application scope of the model.

Some researchers took advantage of distributed technologies and tools to fulfill cooperation in workflow. Daniela Grigori described an approach to support cooperation in a workflow system which was based on the combination of a cooperative transaction protocol and a traditional workflow model, allowing activities to exchange data during execution[9]. Grigori's work emphasized more on flexibility of a cooperative workflow rather than the cooperative workflow itself or cooperation in workflow. Without the distributed framework and technology, the model would not work effectively.

Some researchers used cooperative concept and tool to incorporate cooperation into workflow. Lizhen Cui extended the traditional organizational model with group concept, thus putting up a cooperative work enabled workflow model[10]. But Cui's approach was based on cooperative tools, which had lots of constrains in applicability and maneuverability. Issam Chebbi put up an approach to inter-organizational workflow cooperation, which was inspired by the service-oriented architecture,

consisting of workflow advertisement, workflow interconnection, and workflow cooperation[11]. It was based on adaptability of organization but was also limited by organization and its boundary.

III. ACCESS CONTROL IN COOPERATIVE EDITING SYSTEM

A. Cooperative Editing Flow

The process of cooperative editing is different from individual editing. Generally it has several key phases. Usually a chief editor will first make a sketch of a document and then determine final goal of the document. When the document is set to be coedited enabled, multiple editors can perform editing the document simultaneously. Some will change content of the document, while others may just open and view the document. When an editor finishes editing, he/she will deliver it to the chief editor for reviewing. If the chief editor thinks there need some change, the document will be returned to the editor for revising. After all editors complete editing, the chief editor will set to the document coedited disabled, closing cooperative editing.

The whole process has characteristics of workflow. Incorporating workflow into cooperative editing can make tasks defined in workflow system be automatically executed with predefined rules, which will greatly relieve human efforts, and offer clear control and processing logic, eventually improving cooperative editing effectiveness and efficiency, which is an important facet in cooperative editing.

Cooperative editing allows real time editing by multiple users simultaneously and supports the feature of what you see is what you get. The model we proposed coordinates initialization of cooperation and dynamic check-in and check-out. When cooperation is established, multiple users start cooperative editing communication in a distributed way. The steps involved are given in order as follows.

- Step 1: a user registers a document in CSCE server;
- Step 2: the server automatically assumes the user to be chief editor;
- Step 3: users who take part in editing register in the server;
- Step 4: chief editor makes out workflow for document editing;
- Step 5: chief editor registers workflow in the server and starts workflow engine;
- Step 6: cooperation server is initialized;
- Step 7: cooperation group registers in cooperation server;
- Step 8: workflow engine provides control information from participants to cooperation server;
- Step 9: a participant checks in before taking part in cooperative editing;
- Step 10: cooperation server is responsible for synchronization, data consistency and data concurrency during cooperative editing;
- Step 11: workflow engine is responsible for editing activity flow coordination;
- Step 12: when a participant checks out, workflow engine is responsible for activity state and context processing, version control server is responsible for saving and version coordinating, and cooperation server is responsible for cooperation context.

B. Degrading Control Granularity

Document is treated as a whole object in conventional file system. But it shows many drawbacks in cooperative editing. We take advantage of a more fine-grained object, section, to solve the above problems by degrading control granularity. In our access model[12], all operations are based on section. The definition about section is given as follows.

Definition 1: Section, created by chief editor, is the basic object of access, processing, control and reviewing, and is part of a document.

Hence document is composed by one section or several irrelevant and orthogonal sections with predefined logic, as equation 1.

$$Document = f\left(\sum_{i=1}^n Section_i^v\right) \quad (1)$$

Here $section_1 \cap section_2 \cap \dots \cap section_n = \phi$, v is version v of section i , and $f(x)$ is composition logic.

Element section of Microsoft Word, element section of OpenOffice.org Word and section of Latex are common instances of section mentioned above. Microsoft Word and OpenOffice.org Word both provide sufficient technical documents and APIs for developers, making it easy for implementation.

With element section, we decompose document oriented operation to combinations of fine-grained internal logic unit oriented ones, leading control granularity to content of document. Therefore we have document level control which is responsible for the whole file as an integral part for compatibility with conventional file system and section level control which focuses on content and semantic management which is an extension of conventional file system.

C. Control Model for Cooperative Editing

CSCE involves multiple tasks, multiple users and multiple stages, whose relations are quite complicated. In cooperative editing, there also are various resources, data and editing participants involved in cooperative editing.

So control in cooperative editing is essential all through the whole process. It is necessary to coordinate tasks and resources, and manage access control so as to ensure data consistency and concurrency, and achieve collaboration. Moreover cooperative control in cooperative editing, such as dynamically adjusting editors, concurrency control of editing, collision detecting, avoiding and solving, is also equally important in cooperative editing. We developed a fault-tolerant real-time cooperative editing system[13], thoroughly analyzing the characteristics that a cooperative editing system should have.

It is impossible to manage and successfully fulfill editing task without access control. But neither of conventional access models, RBAC or TBAC, can solve access control problems in CSCE. RBAC is passive and unable to deal with real time dynamic control of CSCE[14]. Although TBAC is active, it is weak in manageability and meanwhile cost inefficient[15]. We propose an activity based cooperative editing access control model[12,16] for access control in ZFlow system.

We use activity to represent the job or processing task in ZFlow. It is a description of a piece of work that forms one logical step within a process. Activities of CSCE during editing are usually iterative, mainly including sketching, editing, revising, auditing and finalizing. An activity requires essential resources to support process execution, such as execution duration. It has different execution states such as ready, waiting, executing, terminated, suspended and canceled. It has properties for access control, such as participants and context. It also has constraints, which must be met during processing, such as deadline and cost.

Primitive operations, which are normally not allowed to be broken down, include workflow defining, cooperation group registering, cooperation initialization, data concurrency maintenance, context consistency maintenance, versioned data maintenance, access control list maintenance, and check-in and check-out management.

It will be a nightmare to manage users without a higher abstract mechanism. Role is therefore used in our control model. The participants within a particular organizational role group can undertake an activity requiring a resource with that set of attributes. There are generally three main roles: chief editor, editor and reviewer. Chief editor is normally regarded as the owner of a document, whose responsibility focuses on management and control, including document management, section management, workflow management, access control list management and other management relating to document. Editor edits the assigned section(s) of the document. Reviewer reads the document and the section(s) and inserts comments to the document and section(s), but is not allowed to make any change to the content.

As document is now composed of sections, a user can have a session role and a document role. A user can have different roles among different sections during an activity or within a section during different activities. For instance, a user U1 is the editor of section S1 and is also the reviewer of section S2 during activity A1; a user U2 is the editor of section S3 during activity A2 and is also the reviewer of section S3 during activity A3.

We need a mechanism to associate participants with a collection of activities. Activity role is a role assumed to a participant to process work and access resources and other objects. The role defines the context in which user participates in a particular process or activity. The role often embraces organizational concepts such as responsibility or authority, and may also refer to other attributes such as resource and time. Activity role is a dynamic concept. Participant will not have the role if the related activity is not initialized and started. Activity role control mechanism achieves both dynamic access control and group management.

IV. COOPERATIVE EDITING PROCESS FLOW MODEL

CSCE is team-worked, computer assisted, distributed, cooperative and interactive. Activities of CSCE during editing, including sketching, editing, revising, reviewing and finalizing, are usually iterative in some phases.

The user first registers a document for cooperative editing in cooperative editing server. The server will assign the user to be chief editor of the document, and require him/her planning out logic structures of the document by plotting out sections. Then the server will initialize section according to the chief editor's settings. The chief editor adds cooperative editing task for each section, and then assigns task to participants who are also managed by the chief editor.

By then, multiple editors can check out sections and proceed to cooperative editing according to the workflow defined previously. An iterative activity of returning will occur if a section fails to go through reviewing and need revising. After all sections have been checked in, document will be again audited as a whole. If the document goes through being auditing, the chief editor will finalize the document; otherwise, the document will be transferred to former step for revising. Figure 1 shows the flow of cooperative editing with workflow.

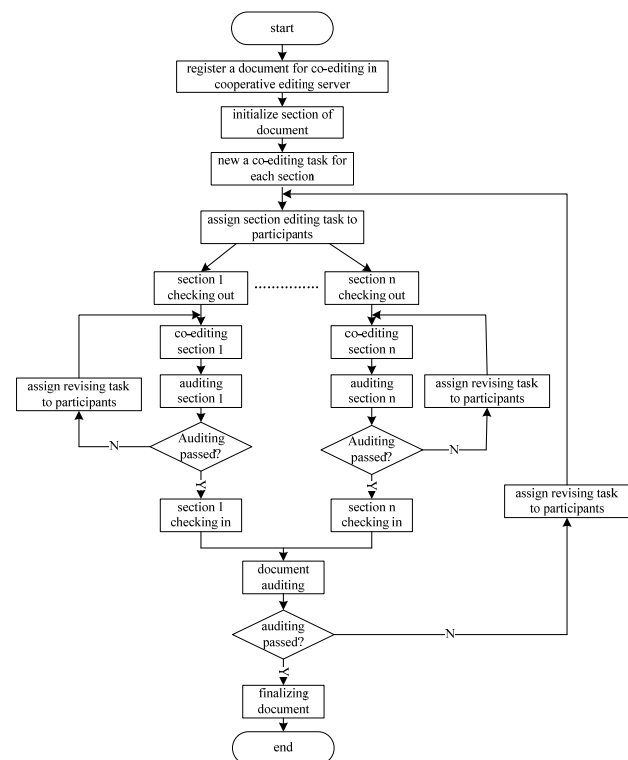


Figure 1. The process flow of cooperative editing with features of workflow. It shows how a chief editor registers a document in cooperative editing server and assigns editing tasks. Editors then perform cooperative editing according to the workflow predetermined. Elements including document and sections are transferred among editors. Eventually the chief editor finalizes the document and completes editing process.

V. ZFLOW ARCHITECTURE

OpenOffice.org is an office suite which can be run in Linux, FreeBSD, MacOS X, Solaris and Windows. OpenOffice.org documents are compatible with most office suites such as Microsoft Office, Sun StarOffice, and Adobe Acrobat. ZOffice is a cooperative editing system that is based on OpenOffice.org. We design a

workflow, ZFlow, which can be used in ZOffice to carry out workflow responsibility.

A. Fundamental Modules in ZFlow

In ZFlow, authorized users can define process for a document and modify the process during editing. ZFlow has two main parts: workflow client and workflow engine. Workflow client is used for defining and revising workflow by editors. Actually it acts as interaction part between computer and user. Workflow engine provides kernel solution for driving cooperatively editing document to transfer among users and determines how elements including document and section routine among users and activities in accordance with workflow activities, constraints, and condition state.

Among the fundamental modules in ZOffice, there are four modules that are most important to ZFlow. They are document flow definition module, workflow engine, editing client, access control module. Figure 2 shows the logic view of ZFlow.

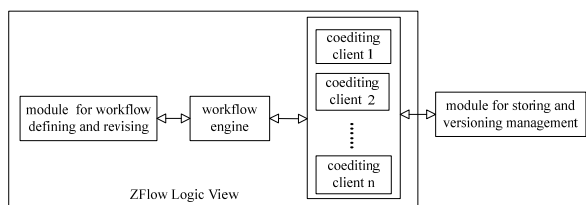


Figure 2. Logic view of ZFlow, which has modules for workflow defining and revising, workflow engine and cooperative editing client.

B. Describing Workflow with ZPDL

There are two typical ways to develop workflow applications. One is to build workflow model with a dedicated language and then to link the tasks. Presently XML-based workflow definition languages, such as XPDL, YAWL and SCUFL, are popular; executable process definition languages based on web services, such as BPEL, BPML, WSFL, XLANG, Wf-XML, SWSL and Job Definition Formats, are also widely used. The other approach is to develop workflow applications by using a programming language with libraries and interfaces, such as Windows Workflow Foundation and Workflow OSID.

We propose an XML-based cooperative editing workflow definition language for ZFlow, which is known as ZOffice Process Definition Language, ZPDL for short. As ZPDL is based on XML, it has characteristics of flexibility, portability and compatibility. ZPDL contains basic information of document and section, workflow control elements and activity information in workflow. It takes both conventional workflow system and cooperative editing system into consideration. By ZPDL, we can describe workflow for cooperative editing system well.

ZFlow has a workflow defining and modifying module, which is a GUI client for user to plan out document editing process. When defining the processing flow, chief editor starts with setting workflow for document level, then sets sub workflow for section level. A series of sub workflows of sections construct cooperative editing framework.

As completing processing flow for a document, workflow client will compose a ZPDL file which comprises information and properties of component relating to workflow. A decomposer will parse the ZPDL file when it is delivered to workflow engine. When a user wants to examine workflow execution state or updates workflow, a composer will construct graphic view for the user.

The following is an example described by ZPDL. ProcessHeader is an element used to represent process head information in ZFlow, which is usually fundamental for a process.

```
<ProcessHeader>
  <Sectionid>001</Sectionid>
  <Docid>01</Docid>
  <Author>0001</Author>
  <Version>1.0</Version>
  <Created>02-12-2009</Created>
  <Description>
    an example showing it how works
  </Description>
  <Duration>15</Duration>
  <DurationUnit>D</DurationUnit>
  <Priority></Priority>
  <ValidFrom>02-12-2009</ValidFrom>
  <ValidTo>02-18-2009</ValidTo>
  <PublicationStatus>
    RELEASED
  </PublicationStatus>
</ProcessHeader>
```

VI. KEY COMPONENTS OF ZFLOW

A. Key Components for Access Control

There are six key components for access control in ZFlow: activity, resource, constraint, participant, context and role. They represent essential information and provide important mechanism for control. The following is brief introduction of the components which have been discussed in our access control model [12].

1) Activity

In ZFlow, activity represents processing task. Component activity represents important control information. It is also one of the fundamental components for flow control. See detailed information in Components for Flow Control.

2) Resource

Component resource stores all available resource that user can get for execution, including time duration for execution, access list that maintains users who can use the resource, and available version that maintains all available versioned sections for checking in and checking out.

3) Constraint

Component constraint represents the set of constraints including deadline of activity, max cooperators and

prerequisite that must be previously executed, and referred objects involved in activity execution.

4) Participant

Component participant indicates users who take part in activity execution and the corresponding roles during activity execution. It is also used in flow control. See detailed information in Components for Flow Control.

5) Context

Component context is used for synchronous and asynchronous control of collaboration and activity state switching of workflow. It has four properties, including current cooperators who are presently taking part in cooperative editing, synchrony which is used for synchronous cooperation control, data consistency and concurrency coordinating, and asynchrony which is used for asynchronous operative control, versioned section check-in and check-out management and users' authority management during offline editing.

6) Role

Component role is used for role and permission control. Administrator can update role set to add, delete and modify role. It has two properties including role name and permission which is assigned according to role.

B. Components for Flow Control

We use nine components to achieve flow control in ZFlow. They respectively are document, section, process, activity, subflow, transition, tasks, processingrate and participant.

1) Document

Component document stores all information of the document. Some key fields of component document are shown in table I.

TABLE I. KEY FIELDS OF COMPONENT DOCUMENT

Key Field	Comments
docid	Unique identifier of the document
docname	Document name
createTime	The time when the document is created by chief editor
version	The version of the document

2) Section

Component section represents information of the section, as shown in table II.

TABLE II. KEY FIELDS OF COMPONENT SECTION

Key Field	Comments
sectionid	Unique identifier of the section
sectionname	Section name
docid	Corresponding document to which the section belongs
version	The version of the section

3) Process

Component process includes all information related to editing process. Every process has its own unique identifier and its process name. A process will be in one of five states: ready, waiting, executing, terminated,

suspended and canceled. There are two types of process: DocumentProcess which represents document editing process and SectionProcess stands for section editing process. We use field docid and sectionid to relate the process to the corresponding document or section. Some key fields of component process are shown in table III.

TABLE III. KEY FIELDS OF COMPONENT PROCESS

Key Field	Comments
pid	Unique identifier of the process
pname	process name
pstatus	Process states
docid	Corresponding document identifier of current related process
sectionid	Corresponding section identifier of current related process
pptype	Process type: DocumentProcess or SectionProcess
activities	Activity set of current process
transitions	Transition set of current process
participants	All participants of current process
constraints	All constraints covered in current process
resources	All resource that could be used in current process
context	Processing context

4) Activity

Component activity includes information of activity in the flow. There are four types of activity: subflow, general, start and end. The activity with value subflow stands for current activity is the activity of subflow; the one with value 'general' represents current activity is a general one, the one with value 'start' means it is the beginning of the flow and the one with value 'end' indicates it is the end of the flow.

Activity has five states which respectively are ready, waiting, executing, terminated, suspended and canceled. As the activity may have multiple executors, we use filed pro_rate to represent current processing rate. We use two joint types, with the value of 'and' and the value of 'xor', to deal with activity convergence and two split types, with the value of 'and' and the value of 'xor', to deal with activity transmission rules. More information is shown in table IV.

TABLE IV. KEY FIELDS OF COMPONENT ACTIVITY

Key Field	Comments
aid	activity identifier
aname	activity name
astatus	activity states
pid	corresponding process identifier
atype	activity type
pro_rate	Processing rate of the activity
jointype	Joint type of convergence
splittype	Split type of divergence
start_time	Start time of the activity
finish_time	Deadline of the activity
transitionRestriction	Restriction for transition
tid	Identifier of transition
finishMode	Finish mode: manual or automatic
participantMode	Execute mode: manual or automatic

5) Subflow

Component subflow is used to represent editing flow control information for sections. Typically each process

has subflow for sections. It also has five states: ready, waiting, executing, terminated, suspended and canceled. We use sectionid and docid to relate the section of the document. Field coopercontext provides context information for cooperative editing and corresponding control. Some key fields of component subflow are shown in table V.

TABLE V. KEY FIELDS OF COMPONENT SUBFLOW

Key Field	Comments
sfid	Unique identifier of the subflow
sfname	Sub flow name
sfstatus	Sub flow states
docid	Corresponding document identifier of current related subflow
sectionid	Corresponding section identifier of current related subflow
coopercontext	Context for cooperative editing

6) Transition

Component transition contains all transition edge information in flow, which is important in flow control. Key fields of component document are shown in table VI.

TABLE VI. KEY FIELDS OF COMPONENT TRANSITION

Key Field	Comments
tid	unique identifier of the transition
from_activity	Identifior of starting node of the current transition edge
to_activity	Identifior of ending node of the current transition edge
pid	Process identifier where the transition resides
condition	Transition condition

7) Tasks

Component tasks includes all information of a collection of tasks. It is also regarded as valid task collection, from which a user can get his/her task. The engine extracts task information, which is related to current flow, from component activity and component participant after the engine is initialized and instanced. Then the engine, after necessary validation, integrates the information to compose a task and add an entry in the component.

All data in the component tasks are maintained by workflow engine itself, not allowing to be modified by others. When any value in the component activity and component participant, such as activity state, changes, the engine will adaptively update the corresponding values in the component tasks, thus keeping consistency among three components. Some key fields of component document are shown in table VII.

TABLE VII. KEY FIELDS OF COMPONENT TASKS

Key Field	Comments
taskid	Unique identifier of the task
aid	Activity identifier
pid	Unique identifier of the process list
sectionid	Section identifier
parptmid	Identifier of activity participant
docid	Identifier of the document
operationlist	All operations that the performer carries out
status	Current task state: ready, waiting, executing, terminated, suspended and canceled

8) Processingrate

Component processingrate shows processing rate of tasks in a group. When a user of the group completes the assigned task, an entry will be added to the component. By querying the component, users can avoid incorrectly submitting the same task several times. If an activity is completed, an corresponding entry will be added to the component. Hence processingrate can also be used to see which the tasks are completed and which are not, therefore being able to see total processing rate. See more information in table VIII.

TABLE VIII. KEY FIELDS OF COMPONENT PROCESSINGRATE

Key Field	Comments
taskid	Unique identifier of the task
sectionid	Section identifier
parptmid	Identifier of activity participant list
docid	Identifier of the document

9) Participant

Component participant includes related information of activity performers. Field operationlist represents all operation that the participant will carry out in the section. See more information in table IX.

TABLE IX. KEY FIELDS OF COMPONENT PARTICIPANT

Key Field	Comments
pid	Unique identifier of the process
parptmid	Identifier of activity performer
partptype	Participant type: individual or group
operationlist	All operations that the participant carries out
aid	Activity identifier

VII. FLOW CONTROL ALGORITHM OF ZFLOW

W.M.P. van der Aalst and K.M. van Hee have discussed workflow and its management[17]. WFMC also presents algorithms for flow control[5]. But there is no existing flow control algorithm for workflow in cooperative editing system. Here we propose a novel algorithm for workflow control in ZFlow, which determines the routine logic. In the algorithm, an activity in workflow is represented by a node.

The chief editor first initializes the edit flow and distributes it. The flow state will be ready at the time, showing it is activated. The workflow engine will automatically set the starting activity to be ready and refresh the tasks list, waiting for the performer to start all the tasks related to the activity. The authorized user gets the corresponding task through control server. If the task is not initialized yet, the user carries out initialization manually. Meanwhile control server will notify the workflow engine to set task state to the value of 'waiting' and then update the active tasks list, showing the activity is now waiting for execution. After the user completes the task, control server will inform the engine of setting the corresponding activity state to value of 'terminated' and refreshing the tasks list, indicating the activity has completed execution.

Each time an activity completes execution, the engine will exam whether its successor node is an ending node which indicates the end of the flow. If the node is not an

ending one but a common one, the engine will automatically set all successor nodes to value of 'ready'. If the node is a subflow, then the engine will initialize the subflow and set the beginning node in the subflow to value of 'ready', showing everything is ready and just waiting for starting execution.

In the case of convergence and divergence, the engine determines when to and how to activate the node by the value from field jointype and field splittype in component activity. The engine locates the next node with value of 'ready' by travelling workflow graph, recording the last one with value of 'ready' in the current travelling as the starting node of next travelling. As soon as the last activity finishes execution, the engine will set the flow to state to value of 'terminated'.

If current node has more than one adjoining nodes, which usually stands for cooperative editing, it will be processed as router; otherwise it will be processed as a general node. If current node has more than one precursor nodes, then the engine will process differently according to the transfer type of current node, and convergence or and divergence, as well as whether the type of current node is subflow or general node. The following is the core part of the algorithm used for cooperative process, showing how to deal with flow control in such case.

```

Get current node's all preceding processed nodes;
IF (current node is and convergent)
{
  IF (the node is a subflow AND it is not started AND
  all its processed preceding nodes)
  {
    Start up all activities of relating subflow;
  }
  ELSE IF (the node is a general node AND it is not
  started AND all its preceding nodes are processed)
  {
    Set the node to be ready for startup;
  }
  Add the node to router queue;
  Continue traversing forward;
}
ELSE IF (the node is and divergent)
{
  IF (the node is a subflow AND it is not started AND
  one of its preceding nodes is processed)
  {
    Start up all activities of relating subflow;
  }
  ELSE IF (the node is a general node AND it is not
  started AND one of its preceding nodes is processed)
  {
    Set the node to be ready for startup;
  }
  Add the node to router queue;
  Continue traversing forward;
}

```

As for non-operative processing, current node will have only one adjoining node. The engine will process according to whether the current node is a subflow or a

general node along with current node's state. The following shows the main process logic of this.

```

IF (the node is a subflow AND it is not started AND
all of its preceding nodes are processed)
{
  Start up all activities of relating subflow;
}
ELSE IF (the node is a general node AND it is not
started AND all of its preceding nodes are processed)
{
  Set the node to be ready for startup;
}
ELSE IF (current flow is terminated AND the node is
not started AND all of its preceding nodes are
processed)
{
  Set the node to be terminated;
}
Add the node to router queue;
Continue traversing forward.

```

VII. CONCLUSION

With the enforcement of globalization of enterprise and organization, demands for worldwide production and service are getting more and more urgent. Systems that support cooperative working will take on irreplaceable effect in many aspects including improving efficiency and reducing cost. It will be a necessary infrastructure in information processing. Applications of cooperative working such as cooperative editing, virtual organization, electronic business, will have deeply impact on our work, study and life.

Cooperative editing is one of the most important applications in cooperative working, and is widely used presently. Workflow supports automation for cooperative editing, greatly relieving human work and offering convenience in management. A system combining cooperation with workflow shows us an approach to solve newly emerging problems in large scale and knowledge intensive editing.

We put up a workflow for cooperative editing. The model, which is based on activity, manages users and resource, considers constraints in editing, controls activity execution, and provides interfaces for cooperation concurrency and consistency.

We apply workflow to ZOffice, which is cooperative editing system and has many key technologies such as activity based access control for cooperative editing as model in access control, a novel algorithm for cooperation, ZOffice process definition language for workflow, and a novel workflow control algorithm. We find ZOffice with ZFlow can deal with cooperative editing well, spare much work in cooperative editing, and has clarity in process logic.

The model introduced in the paper can also be extended to other collaborative systems, such as cooperative programming system.

ACKNOWLEDGMENT

We would like to thank help from CKC Institute of Chinese Information Processing Technologies.

Funding: This work was supported by the "863" National High-Tech Research and Development of China under Grant No.2006AA01Z147, the National Natural Science Foundation of China under Grant No.60673041, the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No.20060285008, and the National Natural Science Foundation of Jiangsu Province under Grant No.BK2008160.

REFERENCES

- [1] CoWord , <http://cooffice.ntu.edu.sg/coword/>.
- [2] CoPowerPoint, <http://cooffice.ntu.edu.sg/copowerpoint/>.
- [3] Jacob Biehl, Mary Czerwinski, Greg Smith, George Robertson, Brian Bailey, "FASTDash: A visual dashboard for fostering awareness in software teams," CHI 2007, Conference on Human Factors in Computing Systems, San Jose, CA, USA, April 28-May 3, 2007, pp.1313-1322.
- [4] Jacob Biehl, William Baker, Brian Bailey, Desney Tan, Kori Inkpen, Mary Czerwinski, "IMPROMPTU: A New Interaction Framework for Supporting Collaboration in Multiple Display Environments and its Field Evaluation for CO-located Software Development," CHI 2008, Conference on Human Factors in Computing Systems, Florence, Italy, April 5-10, 2008.
- [5] Workflow Management Coalition, <http://www.wfmc.org>.
- [6] Toan Nguyen, Vittorio Selmin, "Cooperative Design Workflows for Multiphysics Applications," Cooperative Design, Visualization, and Engineering, LNCS, Springer Berlin/Heidelberg, 2008, pp.281~285.
- [7] I. T. Hawryszkiewicz, "An Ontological Approach for Defining Agents for Collaborative Applications," Web Information Systems Engineering, LNCS, Springer Berlin/Heidelberg, 2005, pp. 81~94.
- [8] Zhilin Yao, Shufen Liu, etc, "An Ontology Based Workflow Centric Collaboration System," Computer Supported Cooperative Work in Design III, LNCS, Springer Berlin/ Heidelberg, 2007, pp.689~698.
- [9] Daniela Grigori, Hala Skaf-Molli, François Charoy, "Adding Flexibility in a Cooperative Workflow Execution Engine", High Performance Computing and Networking, LNCS, Springer Berlin/Heidelberg, 2000, pp.227~236.
- [10] Lizhen Cui, "Research on Cooperative Workflow Management Systems," Computer Supported Cooperative Work in Design, LNCS, Springer Berlin/Heidelberg, 2005, pp.359~367.
- [11] Issam Chebbi, Schahram Dustdar, Samir Tata, "The view-based approach to dynamic inter-organizational workflow cooperation," Data & Knowledge Engineering, No.56, 2006, pp. 139~173.
- [12] Fei Zhu, Qiang Lv, "ACEAC: A Novel Access Control Model for Cooperative Editing with Workflow," 2008 International Symposium on Electronic Commerce and Security, IEEE Press, New York, 2008, pp.1010-1014.
- [13] Shubin Wang, Fei Zhu, Lv Qiang, "Fault-Tolerant Real-Time Cooperative Editing Systems Software," Computer Applications and Software (in Chinese), Vol.23 No.3, 2006, pp.61~63.
- [14] Sandhu R, Coyne EJ, Lfeinstein H, Youman CE, "Role-Based Access Control Models," IEEE Press, Vol.29 No.2, 1996, pp.38~47.
- [15] Thomas R.K, Sandhu R.S, "Task-based Authorization Controls (TBAC): A Family of Models for Active and Enterprise-oriented Authorization Management," Proceedings of the IFIP TC11 WG11.3 Eleventh International Conference on Database Security XI: Status and Prospects, London, UK, 1997, pp.166~181.
- [16] Fei Zhu, Qiang Lv, Yiyong Shi, "Design and Implementation of Access Control in Project Management Platform," Computer Applications and Software (in Chinese), Vol.24 No.9, 2007, pp.105~108.
- [17] W.M.P. van der Aalst and K.M. van Hee, "Workflow Management: Models, Methods, and Systems," MIT Press, Cambridge, MA, 2002.

Fei Zhu was born in Suzhou, China, in 1978. He got his master's degree in 2006, majoring in computer science and technology. He is now a PhD student of Soochow University, studying on bioinformatics and systems biology.