社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－DBS－141 (6)
2007－GN－ 62 (6)
2007／1／25

発言者主導モデルを採用したリアルタイム次世代CMSの開発

岩田明典† 高丸尚教

将来，学習および研究の場でリアルタイムコミュニケーションを活用していくための基礎概念として，発言者主導モデルを提案する．このモデルはCMSが扱うデータとして静的なものだけでなく，ユーザが発言の場に参加するという動的な過程自体をコンテンツ作成に利用するという手法である．コンテンツ作成から作成後の再利用についての柔軟な取り扱いを実現するため，データの公開範囲に注目したデータ管理手法である，発言領域という概念を導入する．コンテンツ作成の包括的アイデアと，データ保護の観点からも有利な基本概念を利用し，次世代CMS開発への基礎構築を行った．

Development of the next generation realtime CMS by presenter driven model

AKINORI IWATA† and HISANORI TAKAMARU

In order to handle both dynamic and static contents with real-time communications in the next generation of CMS, "Presenter driven model" is proposed. In this model,
(1) the difference between "private" and "public" actions/contents is clarified,
(2) the dynamic process which an user takes part in the communication field is handled as the extended contents,
(3) the "unix-like directory" concept is introduced as the intersection between "private" and "public" ones,
(4) advantages of both contents creation and data protection is comprehensively given.

1. はじめに

CMSは非同期にデータを共有するためのポータルとして広く利用されている．我々は，より利便性に長けたCMS開発のためにリアルタイム・セッション機能を加えた包括的CMS環境の必要性を訴える．そこで，本提案モデルでは，講義や会議をモデルとした発言者による共有セッションの管理を軸として，ネットワーク黒板システム(NBS)を試作している．NBSを通した発言は，全てレイヤ処理することで，細かいアクセス制御やSOX法等のセキュリティの要求に柔軟に対応できるよう基礎設計されていることが特徴である．さらに個人のメモ・描画イベントも同列に扱うことでリアルタイムなコンテンツの編集や活用ができ，サーバ・クライアントやP2Pnetworkでも動作し得る統合環境基盤を構築することができる．

2. 将来のCMSに向けて

2.1 現状のCMSが抱える問題点

はじめに，現在のCMS☆について概観し，次に，現状のCMSが扱うデータが，筆者らの考える本当の意味でのリアルタイムなデータを扱うことを目的としたCMSの構築に不十分であることを説明する．

現在，CMSはスケジュールやToDoの管理を行い，グループウェアとして代表的なNotesやXOOPSのようにウェブ上にコミュニティを作成するために記事や掲示板を設置するためのものが主流である．また，ブログは日記にトラックバックの仕組みを設け記事同士のつながりを持たせている．さらに，Wikiのような参加型ウェブ編集サイトや，SNSのようにコミュニケーションを促進し，知り合いを通じて有益な情報を集めることを容易にしたシステムなど多種多様である．様々な形態を採るCMSであるが，以後議論の対象とするのはポータル機能やグループウェアとしてのCMSである．

† 中部大学大学院工学研究科情報工学専攻
Graduate school of Computer Science, Chubu University

☆ Contents management system. 学習用途に特化したCourseware management systemも含める．

これらのCMSに共通するのは扱うデータがテキスト，画像，音声，動画など静的であることだ．アクセスの手段としてはウェブブラウザを用いることが多い．そのため，ウェブブラウザの表現形式に合わせる必要がある．

ここで，筆者の考えるリアルタイムなデータについての考えを提示する．リアルタイムなデータとは，講義や会議といった場で交される議論や，その場でとるメモのような「生」のデータである．このデータを直接扱うことが可能なCMSを開発することによって，従来のCMS上で行われる，サイト上の静的なデータに対する「反応」が主体のコミュニケーション☆1から，時間軸を活用した臨場感のあるコミュニケーション☆2が可能になる．また，このリアルタイムに行われた交信の結果を「公」と「私」のデータに分離する機能を設けることで，それらを活用し易くする他，個人情報保護の観点から，ユーザが所持するデータをユーザ自らがコントロールし易くすることも同時に必要である．

上記のような機能を，既存のCMSに求めると，その要求はもはやCMSの扱う範疇を外れていると言っても過言ではない．しかし，リアルタイム型コミュニケータという単体のアプリケーションで独立したシステムとして構築するより，コンテンツ作成と管理という視点が加わることで，ユーザが参加したコミュニケーションを結果にまとめるだけではなく，その中から有益な情報を得るための手段として活用できるのではないかと考えている．そこで，ユーザの参加という観点について，次に述べる発言者主導モデルというコミュニケーションのモデルを提案する．

2.2 発言者主導モデル

発言者主導モデルとは，オンラインで行われるリアルタイムなコミュニケーションを促進するために，実社会での会議や講義を参考にしている．このモデルの特徴は，発言によって場が進行し，講義や会議など誰もが体験したことのあるコミュニケーションの方法をコンテンツ作成に応用することである．

第1に，実際の会議や講義を考えるための構成要素として，その場に存在する人物や道具を挙げていく．人物面からは，中心となるプレゼンター，司会☆3，その他の参加者，傍観者が挙げられる．道具の面からは，プレゼンターの手元にある資料，全員に資料を説明するためのスクリーン，参加者の手元にあるノートなどが挙げられる．

第2に，道具を用いて会議室または講義室に集う人物の間で行われるコミュニケーションについて考えていく．主となるコミュニケーションはプレゼンター対参加者の資料の提示とそれらの理解に至るプロセスである．プレゼンターは参加者に理解させる内容を，資料の配布または，スクリーンに必要な情報を提示することにより，その意図を伝えるための努力をする．参加者は，プレゼンターの提示する内容を理解するために，ただプレゼンターの発表を聞くだけではなく，資料の一部分に印を付けたり，メモを取るといったように，自らの理解を助けるための方策をとる．参加者が理解を進めるための方策は，このような手元で行うものだけではなく，周囲にいる参加者との相談や雑談など，主のコミュニケーションとは別の小さな範囲でのみ行われる発言にも及ぶ．また，これらのコミュニケーションが突如として，全体のコミュニケーションの場に引き出されることもある．司会に相当する立場の人物は，参加者とプレゼンターの全体への発言や参加者間でのコミュニケーションについて統御する場合がある．これは，全体の進行を妨げないようにするための配慮であったり，また議論を活発にするための図らうといった場面である．議論に参加するメンバーだけで，会議室に入る場合もあれば，公開討論会のように，ある閉じた空間だけではなく，傍観者も議論に参加し得る状況も存在する．つまり，主題が提示されているような会議であっても，その規模は全ての参加者に公開されるという状況から，たった二人で行われる全体から外れた話し合いまで大小は様々である．

第3に，会議や講義が終了し，散会した後のこれらの結果の利用について見ていく．個人の手元に残るのは，講義の過程が記録されたノートと資料，そしてメモである．これらの記録に，もし理解に至るまでの過程が事細かに記録されていれば，後に資料として見返したとき，自分がいつ (When)，どの点を (What)，どのように注目し，理解していったか (How) という一連の流れを再び目にすることで，ただ配布された資料を見るという行為だけでは得られない，より深い理解を得られる上，活用できる範囲を広げることができる．ノートやメモといった個人で作成した文書は会議中に全体に提示されることもあるが，終了後にコミュニケーションの結果を参加者の間で回覧，または一部の譲渡☆4が行われる可能性もある．

第4に，より広域での結果の利用について考える．

☆1 1D, 2D, 3Dのコミュニケーション
☆2 1D, 2D, 3D + 4Dのコミュニケーション
☆3 プレゼンターと同じ場合もある
☆4 例えば，ノートを破って他の参加者に渡す

各々の会議室となる場所はインターネットに接続されていることが必須ではなく，インターネットに接続しない LAN 上で議論を行いたいという場面も想定している．そのように散発的に行われた記録は，個人の環境に残るというだけではなく，別の環境に移動した際に別の会議室での記録と統合することも柔軟に行える必要がある．そのような事例としては，会議を行い議事録にまとめたものを，別の会議で参照し，その内容をプレゼンテーションに引用するといった例を挙げることができる．

以上を踏まえて，このモデルの必要要件について考える．このモデルでは「公」と「私」の範囲が明確にされる必要があるのは明らかであり，「私」の発言が「公」を駆動するための場を提供するという構造を実現し，自由なデータの活用を補助するための可搬性を維持するための仕組みが必要である．

参加者がそれらの情報を，自らがコントロールできるということは個人情報保護の観点からも望ましく，情報の活用の観点からも利点があるということは既に述べた．そのため参加者から見て「公」にあたる空間と「私」にあたる空間が必要となることは自明のことである．この発言を行うための空間のことを発言領域と定義する．

筆者はコンテンツ作成にあたって，リアルタイムに行われる，その過程自体が価値のある情報だと考えている．その条件を満たすためには，発言の場としての講義室または会議室，プレゼンター，参加者，司会の関係，そしてスクリーン，資料，ノート，メモ等の道具，それら全てをシステムとしてまとめユーザに提供する必要がある．言い換えれば，過程をそのまま記録するための仕組みを実現する概念と構造が，発言者主導モデルである．

2.3 マルチプラットフォーム指向

今日の教育現場や研究の場において，コンピューターに搭載されている OS は様々であり，多種多様な OS が混在しているということを前提としなければならない．そのような環境でマルチプラットフォームで動作するソフトウェアを構築するための方法には 2 種類の方法がある．1 つはブラウザをインターフェイスに利用する方法であり，これは既存の CMS 多くが選択している方法である．もう 1 つは，独自の UI を提供し，それをマルチプラットフォーム化するというアプローチである．本研究では，高可搬性を維持し，後に述べるページやレイヤの実現のためブラウザベースではなく，固有の UI を使うモデルで構築する．そのためのサーバ側の実装には Python を，クライアント側の実装には Java を用いた．

サーバの実装として Python を選択した理由は，Python が汎用のプログラミング言語であり，各種ライブラリが充実していること，特に本研究では通信系のライブラリが豊富であることを重視した．Python はオブジェクト指向や整理された明快な構文を持っているなど，初心者から熟練者までそれぞれのレベルにあった開発ができ，ソフトウェアの拡張に関して幅広い労力を取り込める可能性があり，さらに様々なプログラミングスタイルを許容することから柔軟な仕様変更への対応が容易であることが挙げられる．以上の特長を勘案しサーバの実装には Python を利用している．

クライアントの実装として Java を選択した理由は，Java の言語仕様より，標準ライブラリに表現力の高いコンポーネントが備わっていることを重視した．資料，ノート，メモ等ユーザが読み書きできる領域を表現するため，描画には Java2D，画面の構築には Swing という強力なフレームワークを利用できる．さらに今後 2D の画面だけでなく，3D の空間を共有する際には Java3D☆を利用できる．Java もまたオブジェクト指向言語であり，プログラミングに関する資料も多いことから，必要なコンポーネントを別の開発者が全体に影響を与えずに設計・実装することも容易である．

3. 発言領域の概念

発言領域はユーザ間でコミュニケーションを行うためにデータとユーザのアクセス権限を管理するための機構である．ここでは，発言者主導モデルにおいて，複数のユーザ間で共有されるデータをやり取りするための「公」の空間を提供する「チャンネルとセッション」とユーザが管理できる領域を定義する「私」の空間を形成する「ページとレイヤ」という考え方．そして，それらの間で発言をやり取りするフォーマットとしての「メッセージ」という考え方を順に述べていく．

その前に，これらの概念を教育現場での事例を用いて説明していく．この場合，参加者は学生である．学生は講義に参加するためにノートを持っている，そのノートには，講義で配られた資料とその資料に関連した学生自身のメモが書かれている．そのメモとしては資料のキーワードを丸で囲んで強調したり，図を補足するための文書などが書かれるだろう．そのノートは学生の大切な財産であるから，学生が別の講義に持参し必要に応じて参照するということは自然の行為であ

☆ Project looking glass と呼ばれる 3 次元デスクトップ構築プロジェクトに採用されている

る．また，講義中の先生や挙手をした学生の発言は全体に共有されるし，学生と学生が自分の周囲と相談するといった会話は，それらの学生間だけで共有される．

学生が参加する講義が行われる部屋が**チャンネル**である．そして発言や資料が共有される場としての講義自体を**セッション**という．セッションから参加者へ配布される資料は参加者が持参したノートの**レイヤ**に記録され，ユーザ自身がそのノートに記録する書き込みは資料とは別の**レイヤ**に記録される．それらの資料や書き込みはノート内の**ページ**毎に管理される．それぞれの発言や文章を構成しているのは**メッセージ**という単位である．その様子の一例を**図 1** に示す．

以下，それぞれの概念についての詳細を述べる．

3.1 チャンネルとセッション

データを共有するための枠組みとして発言者主導モデルではチャンネルとセッションが必要となる．ここではチャンネル，セッションの順に説明を行う．

チャンネルは実際の発言を共有するための空間であるセッションを束ねるための最上位に位置する．これは実際の講義を例にすると，講師毎に講義室を割り当てることに相当する．オンラインでは，セッションを種類別にカテゴリ分けし，ユーザが参加するセッションを選択する際のガイドとしての機能を持たせることができる．ここで，セキュリティについて考えを巡らせると，多数の互いに関連のないセッションを大規模な環境で同じサーバで運用しているとき，或いは部外者の参加を制限したいという事例が出てくることは明らかである．こういった要求に柔軟に対応するために，標準でアクセス制御のための機構を持たなければならない．

現在，アクセス制御の対象としている項目を以下に示す．これらの認証を必要に応じて組み合わせることで，チャンネルに求められるアクセス制御を詳細に設定することができると考えている．

- 特定ユーザの参加可否
- 外部認証システムを利用したユーザ認証
- 参加ユーザ数の上限
- パスフレーズによる認証

次にセッションの説明に入る．セッションとは，ユーザが参加できる共有スペースであり，ユーザから発せられた発言はこの層を経由して参加者の元へ届けられる．つまり，参加者以外にはその通信内容が配信されることはなく，直接閲覧することはできない仕組みとした．各セッションの管理はシステムを運営するスーパーユーザが任命したチャンネルリーダーが行う．チャンネルリーダーはセッションの予約，凍結，セッションリーダーの任命等のセッション自体の操作とセッションでの参加者の発言領域へのアクセスについて決定する権限を付与される．また，セッションはチャンネルと同等のアクセス権限を個別に設定することができ，上位のチャンネルとは異なる権限を有する必要がある場合にも対応する．

チャンネルとセッションとの違いは，チャンネルはセッションを包含し統御する存在であり，間接的に参加者の交流を補助する概念であるのに対し，セッションは直接参加者同士のコミュニケーションを補助するという点である．

ここで，講義においてある課題が与えられ，グループで協働作業をした後，それを中央のセッションへ送信し参加者全員に提示するという事例を考える．この事例をステップ毎に分割して，チャンネルとセッションの機能を用いて実現する手順を示す．

(1) チャンネルリーダーがセッションの予約，開始
(2) ユーザをグループに割り振り，各グループに含まれるユーザだけに参加を許可するセッションを必要なだけ作成する．
(3) 各セッションで作業開始，終了
(4) 各セッションの作業を中央セッションに送信する

チャンネルには発言の場を用意する毎にセッションが生成されるが，講義では 1 期分の講義をチャンネルと対応させ，開講される複数の講義を収める必要が生じる．そのため，チャンネルの下にファイルシステムに倣ったディレクトリを生成でき，セッションをあたかもファイルの様に配置できるよう設計されるべきである．例えば，「微分積分学」チャンネルに第 1 回，第 2 回，･･･ の講義が存在するという具合である．

ここで (2) のように，中央セッションから派生したセッションを多数作成する場合は，それらのセッションは派生元のセッションのアクセス制限を基本的に踏襲し，各グループ毎にユーザ単位でアクセス制限を掛ける方法が合理的である．よって，派生セッションは同一のディレクトリ下に配置する方針とする．

チャンネルとディレクトリによって，ユーザから見ればセッションを特定することが可能となるため，前述の事例のように，ノートを複数の講義に持っていき，別の講義の情報を参照する際や，講師側で別の講義に使用したものを参照させたい場合に，ファイルシステム上で別のセッションにリンクを張る操作を行うことで，実現することができる．これはチャンネルをネットワーク上のホストと見て，NFS によるファイル共有を行うことと似ている．

このように，チャンネルとセッション，そしてそれら

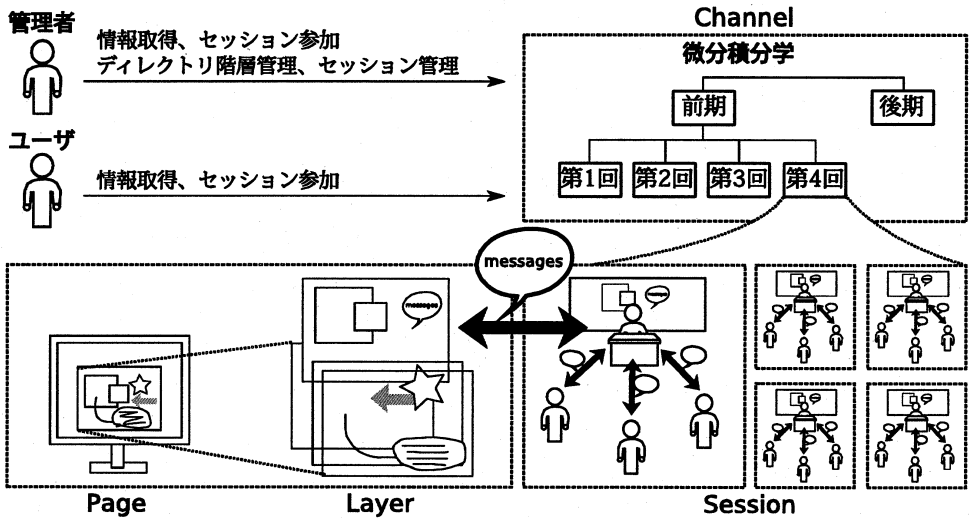

図 1 発言領域の例
Fig. 1 An example for "voice area"

を仲立ちするためのディレクトリという概念を用いることで，コミュニケーションに使うための空間をファイルシステムのように見せることができる．また，要求されるアクセス制限について柔軟に対応することができる機構となっている．

3.2 ページとレイヤ

発言者主導モデルにおいてユーザ側に位置する概念が，ここで説明するページ及びレイヤである．ページ，レイヤの順に説明する．

ページはユーザがセッションに参加するための領域であると同時に，ユーザ個人のメモ等を配置するためのスクリーンの役目を果す．ページには複数の資料を独立して提示することができるよう，1つのセッションにつき複数のページを持つことができるように設計する必要がある．このページに表示するセッションの絵，文書，関連する添付ファイル，メモはそれぞれフィルムのように透明なシート上に配置され，それがページに重なって表示される．このフィルムに相当する領域がレイヤである．チャンネルがセッションを包含するのと似ていて，ページは必要な複数のレイヤを管理する．資料やメモ，そしてテキストを通じた議論の場を形成するため，最低限必要となるレイヤは，2D の画像処理を行うことができるキャンバス，テキストの送受信を行うためのテキストエリアである．

ここで，セッションとページ・レイヤ間のコミュニケーションの指針を示す．セッションからのメッセージは一旦ページが受けとり，メッセージ中に記載されているレイヤの種類を判別した後，ページ内の該当するレイヤへ転送し，レイヤ内で個別の処理が行われる．逆に，レイヤが捕捉したユーザからのアクションはレイヤからページへ一旦転送され，ページで適切な情報に変換した後，セッションへ送信する．

以上で述べた様に，ページとレイヤはユーザ側での処理を分担する機構である．これは，チャンネルやセッションの基本仕様を変えることなく，セッションに必要な表現の幅を広げる拡張を，レイヤの追加によって実現できるということを示唆している．

他にページが持つべき機能としては，ページが統括するレイヤが受けとったメッセージの属性やアノテーションを用いた情報のフィルタリングのほか，レイヤの統合や自由な挿抜ができること等が挙げられる．

3.3 メッセージ

メッセージは，システムとユーザの間でユーザの発言やシステム情報の転送をするフォーマットを指す．

メッセージの本体を構成する情報には，冗長である場合も考えられるが，基本的に1メッセージでレイヤに表示などの影響を与えるために必要な情報を一通り含む方針とした．

具体例を挙げると，ユーザによって資料に引かれたマーカーの線を構成するためには，その軌跡を示す座標のリストと，色，太さ，ユーザ（発言者）の情報が必要である．

4. プロトタイプ

発言者主導モデルを採用した第一のシステムとして，筆者はネットワーク黒板システムを構築した．このシステムではネットワークで情報を授受するためにサーバ・クライアント方式を用いて，Echo サーバを拡張したモデルに基づく単純な構成とした．

4.1 サーバの実装

実装するにあたり，データの送受信のタイミングの制御やリアルタイムコミュニケーションのためのプロトコルを構築する必要があり，通信の層を抽象化するために非同期入出力ライブラリである Twisted[5] を利用している．Twisted は非同期入出力の制御のほか，Python スクリプトから HTTP や FTP 等のサービスと連携させる拡張を施すための手段が提供される．また，データベースへの記録は SQLObject という O/R マッパーを経由し，小規模な利用では SQLite，大規模な利用へは PostgreSQL のような実績のあるデータベース管理システムを利用しメッセージの永続化を行う設計とした．SQL データベースをデータ管理のバックエンドに用いることにより，データアクセスに関するすべてのログを第 3 者へ提供できるように設計した．これによって情報管理の観点から，SOX 法の要求用件への対応を可能とした．

サーバは Python スクリプトで記述されており，上記の必要なライブラリをインストールすることで，個人の PC をサーバとして動作させることが可能となる．ポータブルな実装であるから，ゼミやグループ内での打合せなど小規模のコミュニケーションにもシステムを利用できるという利点がある．

クライアントの要求によって，チャンネルやセッションへの参加のほか，チャンネルやセッションの情報の提供，サーバは発言領域の確保と運用，コンソール経由での管理インターフェイスを提供する．

4.2 クライアントの実装

Java を用いたクライアントアプリケーションの実装を行った．サーバ側との認証等の解釈と講義に見られるようなインタラクティブなやり取りを一通り実現するために必要なレイヤを備えている．具体的には黒板に相当する Drawing Layer，発言を相互に行うための Chat Layer，個人のメモをオーバーレイするための Memo Layer を実装している．このアプリケーションでは，GUI の部品として提供されているタブを利用しページを表現しており，その中で描画やチャットが行えるようそれらのコンポーネントを配置している．また，認証やセッションリーダーからの緊急のメッセージがユーザに感知されやすいよう，必要な場面ではポップアップダイアログを利用して注意を促す．

サーバとの間で送受信及びレイヤ内で利用するメッセージの形式として，プロトタイプ版では敢えてプレーンテキストを利用し，デバッグ時の利便性を優先したが，テキスト形式であるため，今後暗号化するという要求に応えることは難しいことではない．

4.3 チャンネル及びセッションの管理方法

ディレクトリの管理，セッションの予約と管理のための Unix のコマンドに倣ったコマンドラインインターフェイスを備える．

チャンネルの管理 mkdir, chdir, rmdir
セッションの管理 reserve, set
汎用の操作 ls, cd, mv

これらを用いた管理は，管理モードでログインし，そのシェルにこれらのコマンドを入力することで行う．また，コマンドによるオペレーションのため，バッチファイルにこれらのコマンドを列挙することにより，一括処理を行うことができるため，管理作業の効率化を計ることができる．

5. ま と め

本論文では，発言者主導モデルについて提案し，その中でもチャンネル，セッション，ページ，レイヤという場を構成する概念と，その中で発言の伝達を行う媒体としての十分な情報を保持するメッセージという概念を紹介した．そして，提案モデルを取り入れたアプリケーションを実際に制作することにより，今後システムを発展させるための土台を築いた．

6. 今後の展望

本提案モデルを拡張するため方策として，レイヤという構造を最大限に活かし，ユーザレベルでレイヤ内のデータを加工するための方法を拡張する．具体的には，メッセージを外部のデータ処理サービスと連携を可能とするため，プロセス間通信，ソケット通信によるデータの送受信を可能にする拡張を予定している．

参 考 文 献

1) “Sakai Project”, (2005). http://www.sakaiproject.org/
2) Murray G. “Blackboard”, (1997). http://www.blackboard.com/
3) Dougiamas, M. “Moodle”, (2001). http://moodle.org/
4) 垂水 浩幸：グループウェアとその応用, (2000).
5) “Twisted”. http://twistedmatrix.com/