

WebCMS におけるワークフローのパターン化方式

Pattern Approach to Workflow for CMS

猪子 徹†
Toru Inoko

奥野 拓‡
Okuno Taku

大谷 真†
Makoto Oya

1. はじめに

近年、インターネット使用人口の増加に伴い Web サイトは増加、さらに個々の Web サイトのコンテンツも増加している。多くのコンテンツを扱う Web サイトの場合、通常は複数人で共同構築を行う。Web サイトの共同構築を行う場合、ワークフローは重要な位置を占める。Web サイトでは様々なワークフローが使われるが、類似のワークフローが現れることも多い。このような時、類似のワークフローをパターン化できれば、Web サイトの共同構築が軽減される[1]。しかし、最新の WebCMS でもワークフロー作成機能は存在するが、ワークフローのパターン化の方式は示されていない。

本研究では CMS におけるワークフローのパターン化方式を提案する。本方式は、1) パターンの定義、2) パターンの実体化、3) 実システムへのマッピング規則によって構成される。1) はパターンを変数と図式によって定義する方法、2) はパターン定義を CMS 上で実体化する方法、3) では実体化したものをシステムに適用するための規則を定める。それぞれの詳細を提案したのち、CMS として Zope を用い、パターン定義の実験と、定義したパターンの実システムへの適用実験の結果を示す。

2. CMS について

広義の CMS (Contents Management System) では、扱う対象は Web コンテンツに限らず、電子化されていない書類等、様々である。

本論文で扱うのは狭義の WebCMS であり、その定義は、「コンテンツの作成、編集、削除、配信等の各種運用管理を Web 上で一貫して動的に行うことが出来るシステム」である[2]。以下、CMS と表記されたものは狭義の WebCMS とする。

● CMS におけるワークフロー

Zope に代表されるビジネスユースの CMS にはワークフロー作成機能を有するものがある。CMS におけるワークフローの対象はコンテンツタイプである。コンテンツタイプを対象とするのは、CMS におけるワークフローが、同一種類のコンテンツを中心とした作業の流れを示すことを目的としているからである[3]。

● 問題点

CMS で Web サイトを構築すると、異なるコンテンツタイプでも同型のワークフローが現れる。例えば、コンテンツタイプとしてレポートと、日報があるとして、そのワークフローが以下のような場合である。

- ・ 学生がレポートを作成し、助教授がチェック、良ければ教授に提出する。
- ・ 社員が日報を作成し、課長がチェック、良ければ部長に提出する。

この2つのワークフローを見ると、個々に現れる人の種類は違うが、コンテンツタイプを中心とした作業の流れ、即ちワークフローは同型である。同型のワークフローを一つのパターンとして作成できることが必要だが、現状の CMS では、このようなワークフローのパターン化の方式は実現されていない。

3. CMS におけるワークフローのパターン化方式

この章では本論文で提案するワークフローのパターン化方式について述べる。ワークフローのパターン化は以下の3つで構成される。

1. ワークフローパターンの定義
2. ワークフローパターンの実体化
3. 実システムへのマッピング規則

3.1 ワークフローパターンの定義

● アプローチ

ワークフローのパターンを定義するということは、類型化した物(コンテンツタイプ)の状態の変化を定義すると共に、類型化した物と類型化した人の役割との係わりを定義することである。実際にワークフローを割り当てる物、ワークフローを扱う人というのは、各業務において個別のものであり、これはパターン化出来ない。ワークフローのパターン化を行うのは以下の部分である。

- a) 類型化した物の状態がどう変化するか
- b) ある状態において類型化した物の内容に対し、ある役割を持った類型化した人がどういうアクションをとれるか。

ワークフローにおいて、類型化した人の役割が類型化した物の内容をどう扱えるかは、類型化した物の状態によって決まる。

- c) ある役割を持つ類型化した人が類型化した物の状態をどう変化させるか

ワークフローにおいて、ある役割を持つ類型化した人が類型化した物の状態を変化させる。

● ワークフローパターンの定義方法

上で述べた、ワークフローのパターンに現れる要素の定義、定義された各要素の関連性をワークフローパターンを以下の7項目で定義する。

- ・ パターン名
- ・ Transition 名
- ・ State 名
- ・ Role 名
- ・ Contents Permission の定義
- ・ アクティビティ図
- ・ 状態遷移図

各項目が示す内容は表1のようになる。

† 北海道大学情報科学研究科

‡ 公立はこだて未来大学

表1: ワークフローパターンの定義項目

Transition名	物の状態の変化
State名	物の状態
Role名	人の役割
Contents Permissionの定義	物自体への行動
アクティビティ図	人が物の状態をどう変化させるかの流れを示したもの
状態遷移図	人が物の状態をどう変化させるかと、物の状態がどう変化するかの関連を示したもの

定義例を図1に示す。状態遷移図中の○の中には State 名が示されている。○から線でつながっている四角の中にはその状態のとき、ある Role 名を持つユーザが、このコンテンツタイプに内容に対して行える行動を示している。状態間の矢印は Role 名に対応した Transition を示している。

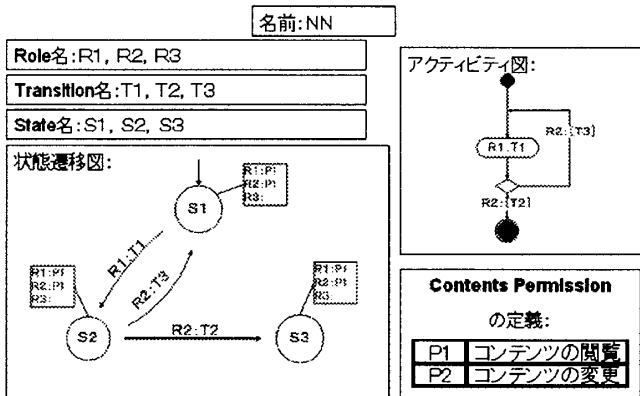


図1: ワークフローパターンの定義図

3.2 ワークフローパターンの実体化

実体化とは、定義図で定義されたワークフローパターンを CMS 上でワークフローとして作成することである。

CMS 上でのワークフローパターンの実体化には以下の7点が必要である。

- ワークフローパターン実体の名前付け
一意の名前を付ける。
- State の実体化
ワークフローパターンの State 名の定義から、各 State の実体化を行う。初期 State がどの State になるのかも同時に設定する。
- Role の実体化
ワークフローパターンの Role 名の定義より、各 Role の実体化を行う。元々 CMS で用意されている Role を使用したい場合は、ここで実体化を行う必要はない。ワークフローパターン特有の Role を用いる場合、Role の実体化を行う。
- Transition の実体化
ワークフローパターンの Transition 名の定義より、各 Transition の実体化を行う。各 Transition がどの State 間の遷移なのかを設定する。Transition の実体を作成するだけで、ここではまだ Role との関連付けは行わない。
- Contents Permission の実体化
ワークフローパターンの Contents Permission の定義から、各 Contents Permission の実体化を行う。

- State と Role と Contents Permission の関連付け
実体化された State, Role と、定義された Contents Permission との関連付けを行う。これにより、Role が各 State でコンテンツ自体をどう扱うことが出来るかが決定する。

- Role と Transition の関連付け
実体化された Role と、Transition との関連付けを行う。これにより各 Transition を用いることが出来るのはどの Role かが決定する。Role のどのような行動により Transition が起こるかも決定する。

3.3 実システムへのマッピング規則

CMS 上で実体化されたワークフローパターンの定義項目と、実システム上のコンテンツタイプ、ユーザグループとの対応を取るため、マッピングを行う。

マッピングするための規則は以下の4点である。

- コンテンツタイプ (又はコンテンツ) 名へのワークフローパターン実体名のマッピング
- ワークフローパターンで実体化された State 名から、実システムの State 表示へのマッピング
- ワークフローパターンで実体化された Transition 名から、実システムでの Transition 表示へのマッピング
- ワークフローパターンで実体化された Role 名の、ユーザグループ名へのマッピング
ユーザグループ (又はユーザ) がコンテンツタイプを扱う時にどういう Permission を持つかを Role を割り当てることでマッピングを行う。

4. 実験

ワークフローのパターン化方式を用い、実際にワークフローパターンの実体化、さらに、実システムへのマッピングを行えるかを実験する。

ワークフローの例として、「学生が絵を描き、担当者がチェック、良ければ閲覧者に公開」といった、作成者、レビュー担当者、閲覧者の3つの役割が現れるパターン (一段階直列レビューパターンと名付けた) での実験を行った。

4.1 ワークフローパターンの定義実験

● 一段階直列レビューパターンの定義

ワークフローパターンの定義からワークフローパターンの実体化を行えるかを実験する。方式に従い、図2に一段階直列レビューパターンを定義した。

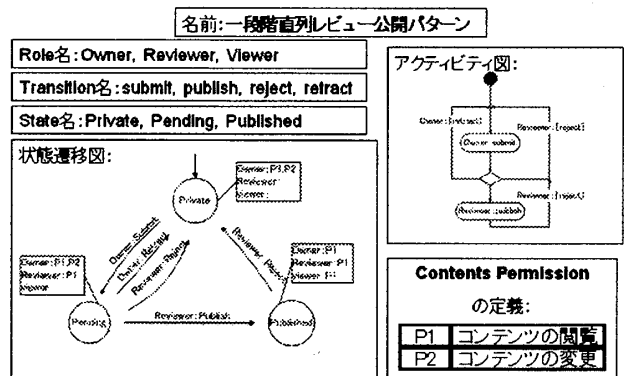


図2: 一段階直列レビューパターンの定義図

各 Role, Transition, State, Contents Permission を定義し、さらにその関連を状態遷移図、アクティビティ図に記述した。

● 一段階直列レビューパターンの実体化

提案した方式の手順に沿って、定義図から実体化を行えるかを実験した。

・ワークフローパターン, State, Role, Transition の実体化

表2: ワークフローパターン, State, Role, Transition の実体化

ワークフローパターンの名前付け	one_level_in_line_review_pattern
stateの実体化	Published, Pending, Private
Roleの実体化	Owner, Reviewer, Member
Transitionの実体化	submit, publish, reject, retract

・Contents Permission の実体化

表3: Contents Permission の実体化

View	コンテンツを見るための権限
Modify portal content	コンテンツを変更するための権限

・State と Role と Contents Permission の関連付け

表4: State と Role と Contents Permission の関連付け

	Owner	Reviewer	Member
Private	View, Modify portal content		
Pending	View	View, Modify portal content	
Published	View	View	View

・Role と Transition の関連付け

表5: Role と Transition の関連付け

	Owner	Reviewer	Member
Private	submit(Pending)	publish(Published)	
Pending	retract(Private)	publish(Published), reject(Private)	
Published	retract(Private)	reject(Private)	

以上の実験の結果、本方式に従うことにより、ワークフローパターンを定義し、実体化することに成功した。

● 実システムへのマッピング実験

提案した方式の手順に沿い、実体化したワークフローパターンの実システムへ適用を実験した。

今回マッピングを行った実システムは「大学授業用コミュニティサイト」であり、現れるユーザグループ、コンテンツタイプは表6のようになる。システムを通じて、授業梗概、プロフィール等の情報を得ることが出来る。

表6: 実システムに現れるユーザグループとコンテンツタイプ

ユーザグループ名	コンテンツタイプ名
instructor	プロフィール
teacher	授業梗概
student	スキル情報
	自由資料
	リンク

一段階直列レビューパターンを適用したコンテンツタイプは以下の2つであった。

- ・講師プロフィール
- ・授業梗概
- ・ワークフローパターンで実体化された State 名から、各コンテンツタイプの実際の State の表示への変換

表7: 各コンテンツタイプにおける State 名の変換

プロフィール		授業梗概	
元のState名	実際のState名	元のState名	実際のState名
Private	プライベート	Private	プライベート
Pending	保留	Pending	保留
Published	公開	Published	公開

- ・ワークフローパターンで実体化された Transition 名から、各コンテンツタイプの実際の Transition の表示への変換

表8: 各コンテンツタイプにおける、Transition 名の変換

プロフィール		授業梗概	
元のTransition名	実際のTransition名	元のTransition名	実際のTransition名
submit	提出	submit	提出
published	公開	published	公開
reject	却下	reject	却下
retract	撤回	retract	撤回

- ・ユーザグループ名への、ワークフローパターンで実体化された Role 名のマッピング

表9: 各コンテンツタイプにおける、ユーザグループへの Role 名割り当て

	プロフィール	授業梗概
Role名	ユーザグループ名	ユーザグループ名
Owner	instructor	teacher
Reviewer	teacher	instructor
Member	student	student

以上の実験の結果、本方式に従うことにより、実体化されたワークフローパターンを実システムへマッピングすることに成功した。

5. まとめ

ワークフローのパターン化方式に沿い、パターンの定義、実体化、実システムへのマッピングに成功した。今後の課題は他の CMS でのパターン化方式の検証、コンテンツタイプのインスタンスに個別にワークフローパターンを割り当てられるよう、Zope のワークフロー機能改善を行うことである。

参考文献

- [1] Russell Nakano, Web コンテンツマネジメントシステム入門—共同作業としての Web サイトの構築と運営, ピアソン・エデュケーション, 2002
- [2] Amos Latteier, Michel Pelletier, Zope 技術入門—Zope の基礎から Web サーバーの開発まで, ピアソンエデュケーション, 2002
- [3] Andy McKay, Alan Runyan, The Definitive Guide to Plone, Packt Publishing, 2004