特別研究報告

題目

ソフトウェア開発過程のオブジェクトを用いたモデル化とそれに基づく開発管理システム
– モデル化とユーザインタフェース部の試作 –

指導教官
井上 克郎 教授

報告者
山本 哲男

平成9年2月21日

大阪大学 基礎工学部 情報工学科

平成9年度 特別研究報告

ソフトウェア開発過程のオブジェクトを用いたモデル化と
それに基づく開発管理システム
– モデル化とユーザインタフェース部の試作 –

山本 哲男

内容梗概

近年，ソフトウェアの開発過程(ソフトウェアプロセス又はプロセスと呼ぶ)を明らかにし，その改善を通じて生産性や品質の向上と開発期間の短縮を図る研究が盛んになってきている．

そのなかで，プロセス記述のためのモデルの研究が数多く行なわれている．しかし，これらは主に開発の作業手順を記述することでプロセスを表現し，作業手順がどこまで終了したかで進捗状況を把握しようとしている．しかし，実際の開発では記述された通りに作業を進めることは難しく，この方法では進捗状況を正しく把握するのが困難であった．

そこで本研究では，開発手順ではなく，開発時におけるさまざまなプロダクトやツール，資源等を属性とメソドを持ったオブジェクトで表現するモデルの提案を行なう．本モデルでは，種々のオブジェクトの状態の変化で進捗状況を知ることができる．

また，本モデルに基く開発管理システムの試作を行なった．本システムは，オブジェクトを保存するリポジトリ部と属性やメソドへのアクセスを制御するユーザインタフェース部から構成される．また，開発作業をオブジェクトに対する作業に限定し，その作業の履歴をとる．各オブジェクトの状態や実行されたメソドの履歴を観察することで，オブジェクトに対する作業状況が分かり，ソフトウェア開発の進捗状況の把握が容易になる．

本報告では主にオブジェクトを用いたプロセスのモデル化手法とhttpを用いたユーザインタフェース部の試作について述べる．

主な用語

ソフトウェアプロセス(Software Process)
ソフトウェア開発管理(Software Development Management)
WWW(World Wide Web)

目 次

1 まえがき **3**

2 提案するモデル化手法 **5**

2.1 オブジェクト ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 5

2.1.1 属性 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 5

2.1.2 メソド ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 5

2.2 オブジェクトの例 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 7

3 開発管理システム **8**

3.1 システムの概要 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 8

3.2 試作 ・・・ 9

3.2.1 リポジトリ部 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 10

3.2.2 ユーザインタフェース部 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 10

4 ユーザーインタフェース部の試作 **11**

4.1 HTML トランスレーター ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 11

4.1.1 index.html ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 11

4.1.2 login.phtml ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 12

4.1.3 process.phtml ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 13

4.2 メソドセンダ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 16

4.3 メソドエグゼキュータ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 16

4.4 システムの動作例 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 17

5 あとがき **20**

謝辞 **21**

参考文献 **22**

付録 **23**

1 まえがき

近年，ソフトウェアの開発過程 (ソフトウェアプロセス又はプロセスと呼ぶ) を明らかにし，その改善を通じて生産性や品質の向上と開発期間の短縮を図る研究が盛んになってきている [8].

これまでに，現実のプロセスのモデル化と記述，よりよいプロセスの探求，プロセス改善手法，プロセスを中心とする開発支援環境の構築，さらにプロセスを実行する人間の特性やコミュニケーションの問題などについての研究が報告されている [6].

そのなかで，プロセス記述のためのモデルの研究が数多く行なわれている [1][2][4]．しかし，これらは主に開発の作業手順を記述することでプロセスを表現し，作業手順がどこまで終了したかで進捗状況を把握しようとしている．しかし，実際の開発ではコーディングとテストを同時に行なったり等の作業の並列実行や単体テストを行なわずに結合テストを行なう等の作業を省略してしまうなど，記述された通りに作業を進めることは難しく，この方法では進捗状況を正しく把握するのが困難であった．

そこで本研究では，開発手順ではなく，開発時におけるソースコードなどのプロダクトや開発者，ツールといった資源等を属性とメソドを持ったオブジェクトで表現するモデルの提案を行なう．本モデルでは，種々のオブジェクトの状態の変化で進捗状況を知ることができる．

属性とはそのオブジェクトの情報・状態を表すものであり，メソドとはオブジェクトを変化させるもので，また変化させる唯一のものである．属性を参照する事でそのオブジェクトの実世界の情報を得，メソドを実行させる事で実世界の対する操作を行なう．

また，本モデルに基づく開発管理システムの試作を行なった．本システムでは，開発作業をオブジェクトのメソドを実行させることで行ない，また，その作業の履歴をとる．履歴を観察することでソフトウェア開発の進捗状況の把握が容易になることが期待される．

本システムはUNIX 上で試作を行ない，オブジェクトを保存するリポジトリ部と属性やメソドへのアクセスを制御するユーザインタフェース部の二つの部分から構成される．

ユーザインタフェース部は，HTML トランスレータ，Web ブラウザ，http サーバの開発者にオブジェクトの属性・メソドを見せるための部分とメソドセンダ，メソドエグゼキュータの開発者の計算機上でのツールを実行させる仕組みを行なう部分から構成される．

Web ブラウザ，http サーバは既存のものを使用し，HTML トランスレータはPHP/FI というHTML ファイル埋め込み型のプログラム言語 [11] を用いて実装を行なう．Web ブラウザからオブジェクトの属性の参照やメソドの実行を行なうようにする．Web ブラウザがオブジェクトについての状態の要求をすると，HTML トランスレータは，オブジェクトの属性一覧，メソド一覧をHTML で記述された文章に変換し，Web ブラウザへその文章を返す

仕組みとする．

メソドセンダ，メソドエグゼキュータはC言語で実装する．メソドセンダはリポジトリのある計算機で動作し，メソドエグゼキュータは開発者側の計算機で動作させる．開発者が作業を行なう過程でツールの起動が必要な場合，メソドセンダがメソドエグゼキュータへツールの起動を依頼を行なう．

本報告ではこれ以降，2章で提案するモデル化について述べる．3章で今回試作した開発管理システムの概要を述べる．4章で試作した開発管理システムのうちユーザインタフェース部の実装について述べる．

2　提案するモデル化手法

本章では，ソフトウェアプロセスのオブジェクトを用いたモデル化について述べる．

提案するモデルではソフトウェアプロセスはオブジェクトの集合で表される．オブジェクトの状態の変化させることにより，ソフトウェア開発を進める．そして，その状態の変化を観察することで進捗状況を判断する材料となる

2.1　オブジェクト

ソフトウェアの開発過程における，仕様書，ソースコードなどのプロダクトや開発者，ツールなどの資源をオブジェクトとする．オブジェクトOは，属性の集合Aとメソドの集合Mの2字組(A, M)である．オブジェクトには一意に認識するための名前が付けられている．

2.1.1　属性

属性$a \in A$は属性名tと属性値vの2字組(t, v)である．属性はオブジェクトについての情報・状態を表すものである．属性名は一つのオブジェクトにおいて一意に定められる，つまり任意のオブジェクト$O = (A, M)$において，

$\forall t\ [\forall v, v'\ (t, v) \in A$かつ$(t, v') \in A \rightarrow v = v']$

である．

属性名のうち，“elements” という属性名はその属性値として他のオブジェクトの名前のリスト“$O_1, O_2, .., O_n$” を取る．あるオブジェクト$O = (A, M)$の属性として

$(\text{“}elements\text{”}, \text{“}O_1, O_2, .., O_k\text{”}) \in A$があるとき，$O = \{O_1, O_2, .., O_k\}$と書く．

このとき，$O_i\ (1 \leq i \leq k)$はOの要素である，あるいはOはO_iを要素として持つと言い，$O_i \in O$と書く．更に，オブジェクトO, O'について，O'が要素として持つ全てのオブジェクトをOが要素として全て持つならば，OはO'を含む，あるいはO'はOに含まれると言い，$O' \subseteq O$と書く，つまり

$O' \subseteq O \Longleftrightarrow \forall O_i\ \ O_i \in O' \rightarrow O_i \in O$

である．

2.1.2　メソド

メソドはオブジェクトに対して許された操作である．全てのオブジェクトの集まりを$\mathcal{O}$としたとき，メソド$m \in M$とは，メソド名sと，$\mathcal{O}$から$\mathcal{O}$への写像fの2字組(s, f)である．メソド名は一つのオブジェクトにおいて一意に定められる，つまり任意のオブジェクト$O = (A, M)$について

$\forall s\ [\forall f, g\ (s, f) \in M$ かつ $(s, g) \in M \rightarrow f = g]$

である．

メソドとして新たなオブジェクトを生成する場合がある．この時，新しく作られたオブジェクトは作り手となったオブジェクトの要素となる，つまり作り手のオブジェクトの属性 “elements” の値であるオブジェクト名リストに新しいオブジェクトの名前が加えられる．

$O_i = (A_i, M_i) \in O = (A, M)$

のとき，

$\forall t\ [(t, v) \in A \rightarrow (t, v') \in A_i]$

かつ

$\forall s\ [(s, f) \in M \rightarrow (s, f') \in M_i]$

であるとする．これをO_iがOの属性名とメソド名を継承すると言う．これにより，あるオブジェクトOからオブジェクトO'が生成されると，Oの属性名とメソド名をO'は継承することになる．

2.2 オブジェクトの例

ソフトウェア開発における，ある「ソースコード」をオブジェクトとして捉えたとすると，その属性・メソドは以下のようなものとなる．

属性

属性名	属性値
ファイル名	/project/foo.c
オブジェクトコード	/project/foo
変更回数	5
変更可能者	yamamoto
最終変更日	1997/2/5
コンパイル回数	3
コンパイル可能者	yamamoto
参照回数	9
参照するオブジェクト	デザイン

メソド

メソド名	内容
変更する	エディタ を起動
コンパイルする	コンパイラを起動
参照する	ビューアを起動

3　開発管理システム

本章では，前に述べたモデルに基づくソフトウェア開発管理システムについて述べる [7].

3.1　システムの概要

本システムは，ソフトウェア開発の環境を提供し，このシステムを用いソフトウェア開発を行なうことにより，開発作業の進捗状況の把握を容易にすることを主目的としたものである.

リポジトリとはオブジェクトの集合，メソドの実行履歴を蓄えておくための場所である. このリポジトリがある計算機のことをサーバ，開発者が作業を行なっている計算機のことをクライアントと呼ぶ (図 1).

開発者はオブジェクトの属性を参照することで，現在のオブジェクトの情報・状態を得る．そして，開発はオブジェクトのメソドを実行することで行なう．またすべての開発過程をメソドの実行という形で限定する．そうすることで，メソドの実行履歴を観察することで作業の現状把握を行なうことが出来る.

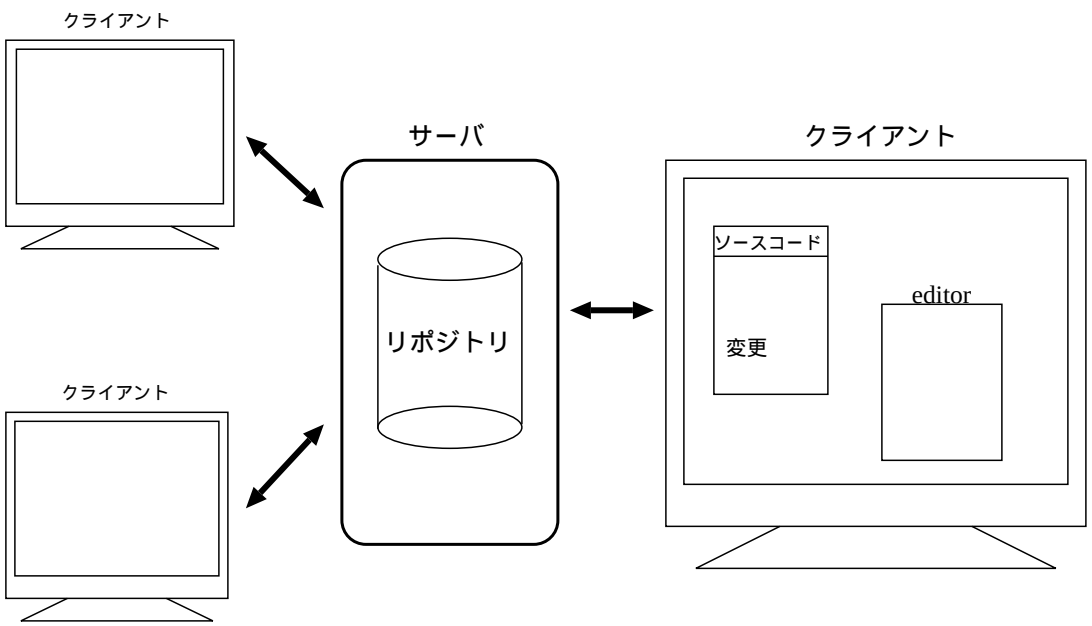

図 1: システムの概要図

3.2 試作

本システムは，UNIX 上で試作を行なった．システムは図2に示すようにリポジトリ部とユーザインタフェース部の 2 つの部分で構成されている．

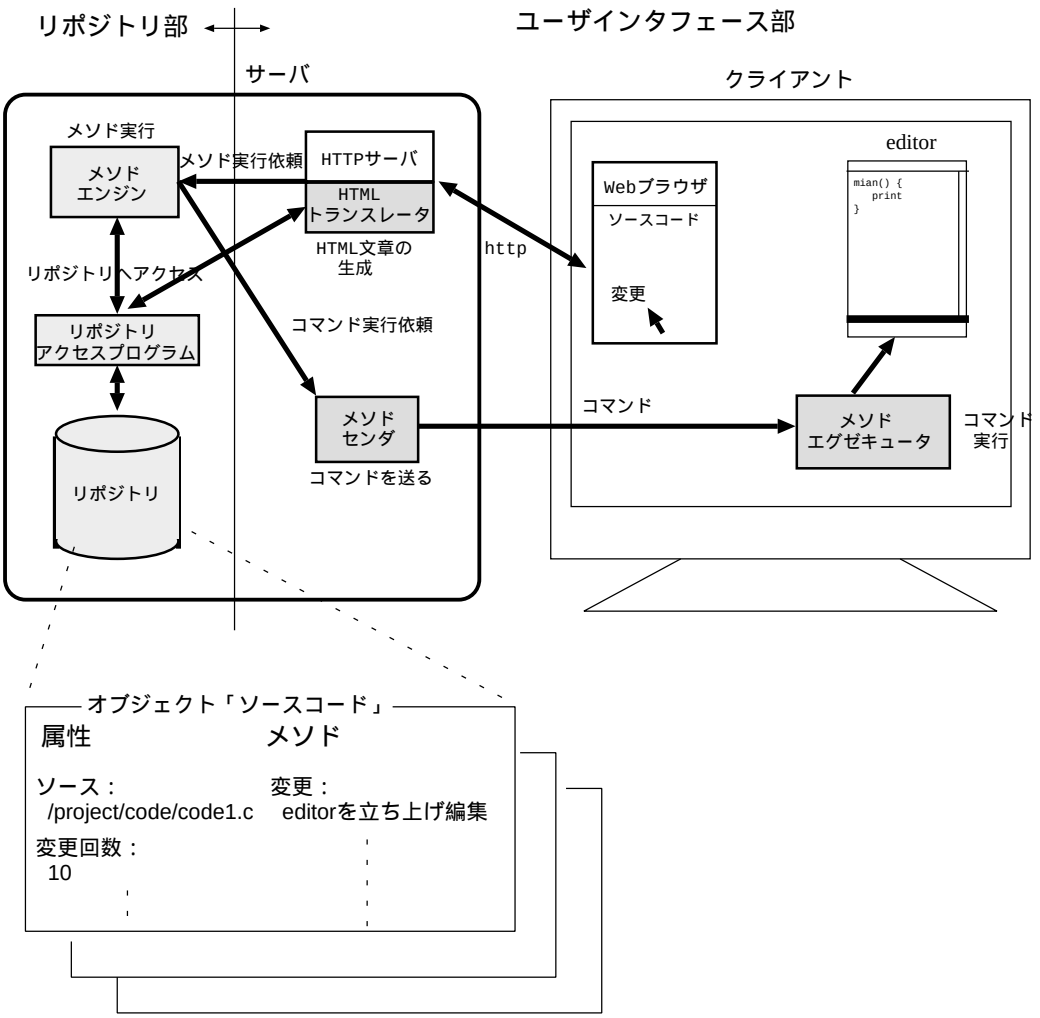

図 2: システムの構成図

3.2.1 リポジトリ部

リポジトリ部はオブジェクトの集合，メソドの実行履歴を蓄えておくための場所であるリポジトリ，そのリポジトリを操作するためのプログラムであるリポジトリアクセスプログラム，メソドの実行を行なうメソドエンジンからなる．詳細は [5] を参照されたい．

3.2.2 ユーザインタフェース部

ユーザインタフェース部は HTML トランスレータ，Web ブラウザ，http サーバの開発者にオブジェクトの属性・メソドを見せるための部分とメソドセンダ，メソドエグェキュータのクライアント上でのツールを実行させる仕組みを行なう部分から構成される．

各オブジェクトには，一意に定まる URL(Universal Resource Locator) が与えられる．この URL は，その URL と対応するオブジェクトの属性名と属性値の一覧・メソド名の一覧が HTML で記述された文章を示している．したがって，開発者は Web ブラウザにその URL を与えるとでオブジェクトの情報を得ることが出来る．

サーバとクライアントの間での HTML 文章の転送には http を用いるので http サーバをサーバ上で動作させておく．

リポジトリにあるオブジェクトは HTML で記述されていないので，各オブジェクトを HTML で記述された文章に変換する必要がある．この役割を果たすのが HTML トランスレータである．HTML トランスレータは，Web ブラウザから HTML 文章を要求された時に http サーバから呼び出される．

また，Web ブラウザ上でメソド名を選ぶことでメソドエンジンが起動されメソドの実行を行なう．

また，メソドを実行するときに開発で使用するツール (エディタなど) をクライアントで起動するなどの記述がある場合，メソドエンジンはそのツールの起動依頼をメソドセンダを呼び出し行なう．呼び出されたメソドセンダは，クライアントで動作しているメソドエグゼキュータにコマンドを実行してもらうように依頼する．このため，開発を行なう前にクライアント上でメソドエグゼキュータを既に実行させておく必要がある．

4　ユーザーインタフェース部の試作

ユーザインタフェース部の試作を行なうにあたり，HTML トランスレータ，メソドセンダ，メソドエグゼキュータを試作し，Web ブラウザは Netscape Navigator[9] 等の既存のものを用い，http サーバは Apache[10] という既存のものを用いた．

オブジェクトの属性を参照したり，メソドの実行はすべて Web ブラウザから行なう．

Web ブラウザを用いることで，開発者は Web ブラウザの操作を知っていれば新たにシステムの操作を覚える必要がなく，開発時の負担を減らすことが期待される．

4.1　HTML トランスレーター

PHP/FI という HTML ファイル埋め込み型のプログラム言語を用いて実装を行なった．HTML 埋め込み型を用いることにより，リポジトリを処理する部分と HTML で書かれたページを出力する部分とを共存することができ，処理が容易となっている．

また，Apache という http サーバを使用することで，PHP/FI のスクリプトを処理する機能を http サーバに付加できることで速度向上が得られる．

HTML トランスレーターは以下の 3 つのファイルから構成されている．

4.1.1　index.html

ユーザ名とパスワードを入力するためのフォームを出力する (図3)．入力されたユーザ名をパスワードは，login.phtml に渡され，そこで正しいパスワードかを判別している．

開発者がシステムを使用するにはユーザ認証を行なわなければならず，まず，このファイルを含む URL をサーバの http サーバに要求しなければならない．

URL の例としては

http://www.foo.or.jp/index.html

のようなものになる．

図 3: ログイン画面

4.1.2 login.phtml

ユーザ認証を行なう部分である．前の index.html で入力されたユーザ名とパスワードがここで確認され，正しいユーザかどうか判別している．もし，正しいと確認されれば，これ以降の Web ブラウザで操作は，そのユーザ名で表されるユーザが実行しているものとし，システムの使用を出来るようにする．

この認証の際に，HTTP Cookies[3] を用いる．これを用いることで Web ブラウザにユーザ情報を記憶させ，一度ブラウザを終了させたとしても，毎回ログインする必要がなくなる．

ユーザ名が “yamamoto” で，HTML トランスレータのあるディレクトリが “/prcoess” であるとした時，どのように cookie が生成されるかを以下で説明する．

1. パスワードが一致すると，login.phtml は

```
Set-Cookie: user=yamamoto; path=/process
```

というヘッダを Web ブラウザに送る．

2. 以降 Web ブラウザは，このサーバに “/process” を含む URL の要求を行なうと，

```
Cookie: user=yamamoto
```

というヘッダを付けて URL の要求を行なう．

3. HTML トランスレータはこのヘッダを認識し，ユーザは “yamamoto” であるとしてユーザ認証を省くことになる．

パスワードファイルは，login.phtml と同じディレクトリにある，“.passwd” という名前のファイルに保存する．パスワードファイルの書式は以下の通りである．

```
ユーザ名:暗号化されたパスワード
```

パスワードファイルの例

```
yamamoto:AR4T0Lh6ZaNTU
fujiwara:mK7X30mTr9qBs
```

4.1.3 process.phtml

HTML トランスレーターの本体部分にあたる部分である．リポジトリを検索し，指定されたオブジェクトの属性一覧・メソド一覧を得，HTML で記述された文章に変換する (図4)．

process.phtml の後に以下の引数が指定できる．

- mono
  オブジェクト名を指定する．必須であり省略することは出来ない．

- method
  メソド名を指定する．この指定がなければ，オブジェクトの属性一覧・メソド一覧が表示される．指定されれば，メソドの実行を行なう．ただし，メソド一覧で，あるメソドを選べばこの引数が指定された URL を指定することになるので．ユーザはこの引数を明示的に付ける必要はない．

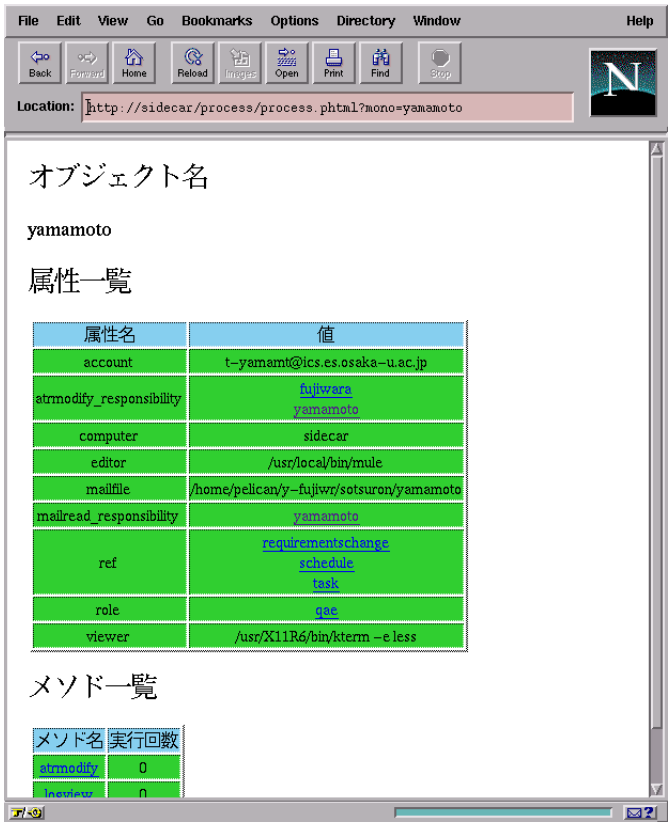

図 4: ブラウザの例

例えば，オブジェクト「sourcecode」の属性一覧・メソド一覧を見たい場合は，URL として

http://www.foo.or.jp/process.phtml?mono=sourcecode

を要求すれば良い．

このURL を要求したときの流れは以下のようになる．

1. ヘッダに cookie として user が設定されていなければ，システムを使用することは出来なず，ログインするようにという文章を Web ブラウザ側に返す．

2. user が設定されいれば，“mono” で指定されたオブジェクト (この場合は “sourcecode”) の属性一覧・メソド一覧をリポジトリアクセスライブラリを用い取得する．

3. 属性値がオブジェクト名であれば，その属性値を選ぶことでそのオブジェクトを指すURL を要求するようにする．属性値が “binarycode” であった場合，以下の URL とな

る．

http://www.foo.or.jp/process.phtml?mono=binarycode

4. メソド名を選ぶことで，そのメソドが実行されるようにする．メソド名が “modify” であった場合，URL は以下となる．

   http://www.foo.or.jp/process.phtml?mono=sourcecode
   &method=modify

5. 以上のようなことをHTML で記述し，その文章を Web ブラウザ側へ返す．

オブジェクトとして “sourcecode”，メソド名として “modify” というメソドの実行を行なう場合，次のようなURL の要求をする．

http://www.foo.or.jp/process.phtml?mono=sourcecode
&method=modify

このURL を要求したときの流れは以下のようになる．

1. ヘッダに cookie として user が設定されていなければ，システムを使用することは出来なず，ログインするようにという文章を Web ブラウザ側に返す．

2. user が設定されいれば，“mono” で指定されたオブジェクト，“method” で指定されたメソド名，利用しているユーザ，クライアントのホスト名を引数としてメソドエンジンを起動する．

4.2 メソドセンダ

C 言語を使い実装を行なった．通常はメソドエンジンから起動される．

使用方法：
sender hostname command ...

hostname: コマンドを送るホスト名
command: 実行させたいコマンド

実行例：
sender sidecar mule file.c
ホスト名 “sidecar” で “mule file.c” というコマンドを実行させたい場合．

起動した後の処理の流れは以下のようになる．

1. 引数に指定されたホスト名とTCP コネクションをはる．
2. 指定されたコマンドの最後に改行をつけ送信する．
3. すべてのコマンドの文字列を送信した後，改行のみを送りコマンドを全て送信したことを示す．
4. メソドエグゼキュータからコマンドを実行後の終了ステータスとコマンドを実行したときの標準出力と標準エラー出力が返ってくるので待つ．
5. 返って来れば，その文字列を標準出力へ出力する．

4.3 メソドエグゼキュータ

C 言語を使い実装を行なった．引数はなく，クライアント側で実行する．
起動した後の処理の流れは以下のようになる．

1. デーモンとして動作し，常にメソドセンダからのTCP コネクションが張られるの待っている．
2. メソドセンダからTCP コネクションが張られると，メソドセンダから送られてくるコマンドを受け取る．受け取った文字列の中で改行はコマンドの区切りを示す．
3. コマンドを実行する．

4. コマンドを実行後の終了ステータスとコマンドを実行したときの標準出力と標準エラー出力をがメソドセンダへ送信する．

5. コネクションを閉じる．

4.4 システムの動作例

あるオブジェクト “sourcecode” のメソド “modify” を実行したときのシステムの動作を説明する (図5)．

1. オブジェクト “sourcecode” に対応した URL を Web ブラウザを用い，http サーバに要求する．

2. HTML トランスレータはオブジェクト “sourcecode” の属性一覧，メソド一覧を HTML で記述された文章に変換する．Web ブラウザでは (図6) のように見える．

3. http サーバは，HTML 文章をクライアント側の Web ブラウザに送る．

4. 開発者は “sourcecode” のページのメソド “modify” を選ぶ．

5. Web ブラウザは，メソド起動ための URL を http サーバに要求する．

6. HTML トランスレータはメソドエンジンに “modify” の実行依頼をする．このとき，ユーザ名・クライアントのホスト名も引数として渡される．

7. メソドエンジンはリポジトリを検索し，オブジェクト “sourcecode” のメソド “modify” の内容を取り出し，実行する．

8. もし，メソドにクライアントで実行するコマンドの記述があれば，メソドセンダにコマンドの実行依頼をする．

9. メソドセンダは，クライアント上のメソドエグゼキュータにコマンドを送信する．

10. メソドエグゼキュータはコマンドを実行する (図7)．

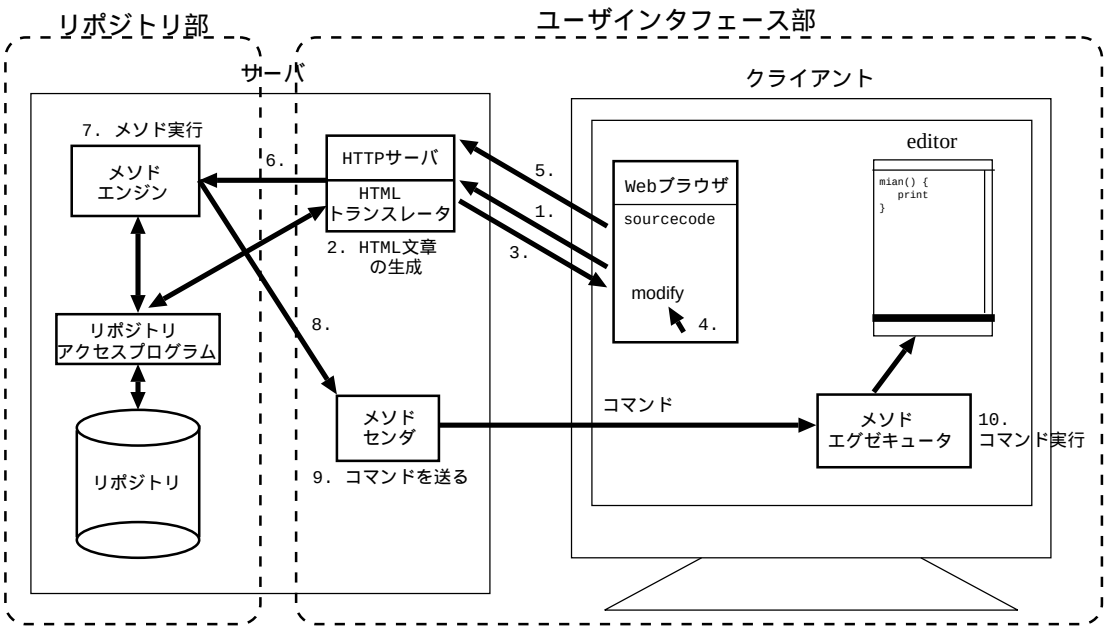
リポジトリ部
ユーザインタフェース部
サーバ
クライアント
7．メソド実行
メソド
エンジン
6.
HTTPサーバ
HTML
トランスレータ
2．HTML文章
の生成
5.
1.
3.
Webブラウザ
sourcecode
modify
4.
editor
mian() {
print
}
リポジトリ
アクセスプログラム
8.
メソド
センダ
9．コマンドを送る
コマンド
メソド
エグゼキュータ
10.
コマンド実行
リポジトリ

図 5: 実行全体図

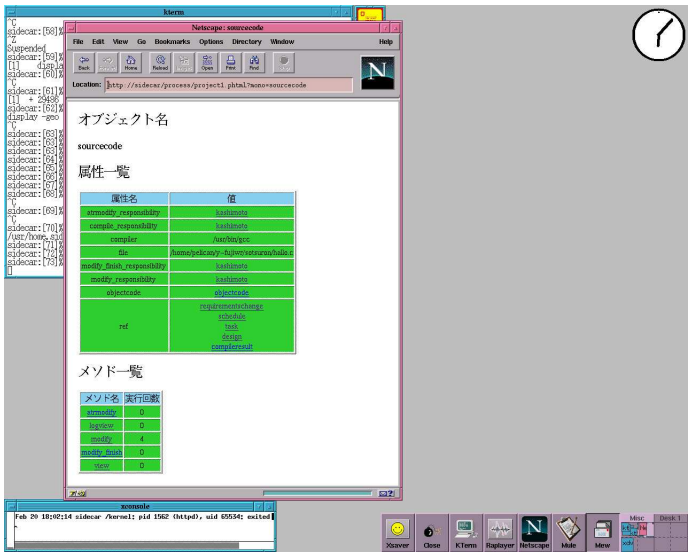

図 6: 実行例 1

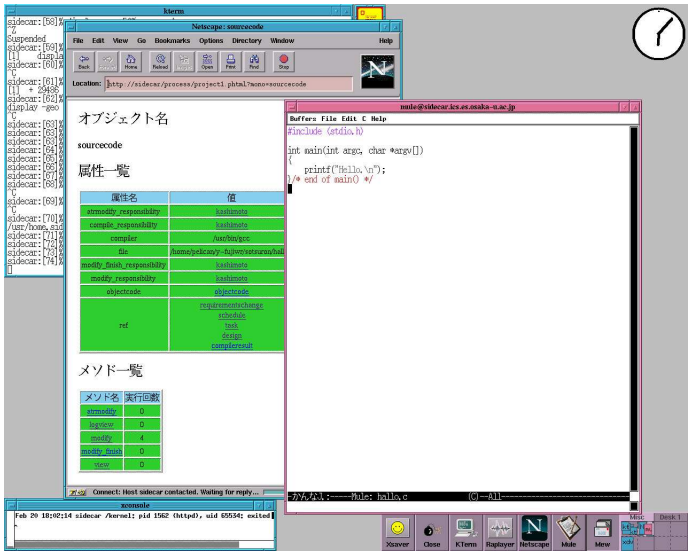

図 7: 実行例 2

5　あとがき

開発手順によらず，開発時におけるプロダクト・ツール等をオブジェクトとして捉え，それを用いたプロセスのモデル化を行なった．

また，そのプロセスモデルに基づくシステムにおいて，ユーザインタフェース部の試作を行なった．

本モデルではオブジェクトの概念を導入することでプロダクトや資源，あるいはそれらの集合といったものを統一的に扱う．また，オブジェクトに対する操作をメソドという形で限定するので，作業履歴が自動的に記録しやすい．

今回，ユーザインターフェースにWebブラウザを用いることで，

- リポジトリのあるサーバ上で作業を行なう必要はなく，開発者は各計算機上で開発が行なえることで，分散化に対応する．
- 既存のWebブラウザを用いることが出来，システムの導入が容易にに実現できる．
- 今までにWebブラウザを使用した経験があれば，使用法などを覚える必要なく，すぐに使うことが出来る．
- 既に多くのプラットホーム上でWebブラウザは存在しており，メソドエグゼキュータのみを移植することで，クライアント側の対応プラットホームが増える．

という利点がが挙げられる．

また，httpサーバとしてApache，処理するプログラム言語としてPHP/FIを用いることで，Webブラウザから URLの要求を受けたhttpサーバは，オブジェクトの属性一覧，メソド一覧をHTMLで記述された文章に変換する時に，変換を行なうためのプログラムを起動せず，httpサーバ自身が処理することで高速性が望める．

今後の課題としては，有効性を実験的に評価することが挙げられる．また，メソドエグゼキュータを他のプラットホームに移植することで，クライアントのUNIX以外でのプラットホームの対応も挙げられる．

謝辞

本論文を作成するにあたり，常に適切な御指導を賜わりました大阪大学 基礎工学部 情報工学科 井上 克郎 教授に心より深く感謝致します．

本論文の作成において，適切な御指導および御助言を頂きました 奈良先端科学技術大学院大学 情報科学センター 飯田 元 助教授に深く感謝致します．

本論文の作成において，適切な御指導および御助言を頂きました大阪大学 基礎工学部 情報工学科 楠本 真二 講師 に深く感謝致します．

最後に，その他様々な御指導，御助言等を頂いた大阪大学 基礎工学部 情報工学科 井上研究室の皆様に深く感謝致します．

参考文献

[1] Bandinelli S.C., Fuggetta A. and Ghezzi C.: “Software Process Model Evolution in the SPADE Environment”, IEEE Transaction of Software Engineering, Vol.19, No.12, pp.1128–1144 (1993).

[2] Curtis B., Kellner M. and Over J.: “Process Modeling”, Communication of the ACM, Vol.35, No.9, pp.75–90 (1992).

[3] David M.K. and Lou M.: “HTTP State Management Mechanism”, RFC2109 (1997).

[4] Fernström,C.: “PROCESS WEAVER: Adding Process Support to UNIX”, Proceedings of the 2nd International Conference on the Software Process, pp.12–26 (1993).

[5] 藤原豊: “ソフトウェア開発過程のオブジェクトを用いたモデル化とそれに基づく開発管理システム –リポジトリ部の試作と共通例題への適用–”, 大阪大学基礎工学部情報工学科特別研究報告 (1997).

[6] 落水浩一郎: “ソフトウェアプロセスに関する研究の概要”, 情報処理, Vol.36, No.5, pp.379–391 (1995).

[7] 大下誠, 松下誠, 飯田元, 井上克郎: “オブジェクトに着目したソフトウェア開発管理システムの提案”, 情報処理学会第54回全国大会発表予定.

[8] Osterweil,L.: “SOFTWARE PROCESSES ARE SOFTWARE TOO”, Proceedings of the 9th International Conference on Software Engineering, pp.2–13 (1987).

[9] “Welcome to Netscape”, http://home.netscape.com/

[10] “Apache HTTP Server Project”, http://www.apache.org/

[11] “PHP/FI Version 2.0”, http://www.vex.net/php/

付録

付録として以下のプログラムリストを載せる.

- HTML トランスレータ
  - index.html
  - login.phtml
  - process.phtml
- メソドセンダ
  - sender.c
- メソドエグゼキュータ
  - executord.c