ユビキタスユーザインタフェースミドルウェア (4)
– 異種端末間での作業状態の引継ぎ –

轟木 伸俊† 北村 操代† 石原 鑑†
† 三菱電機株式会社 先端技術総合研究所

1 はじめに

筆者らは，PC の Web ブラウザや携帯電話等の複数異種の端末から利用できるアプリケーションの構築・実行環境の開発を進めている [1]．本環境では，端末毎に描画能力が異なることを考慮し，端末毎の描画能力に応じて異なる画面フローを提供する．画面フローは状態遷移図で定義される．本環境上で，アプリケーション作業の中断後，再開した端末が中断前の端末と異なる種類の場合に，作業の引継ぎ方法が課題となっていた．

本稿では，各端末の状態遷移図間で共通に使用されるアクションを利用した作業の引継ぎ手法を提案する．また，提案手法を用いたシステムを試作し，本手法の有効性を確認する．

2 異機種間での作業状態の引継ぎ

本環境は，時間・場所に依らずに，作業を遂行できる実行環境を提供することを目的としている．すなわち，作業を開始してから完了する迄に，使用する端末が異なる種類のものに変更されてもよく，また，使用端末を変更するために，ユーザが端末間の同期処理などの特別な作業を必要としない．例えば，社内のＰＣで作業を開始し中断した後，出張時に社外で携帯電話を使用して，中断した時点から作業を継続することができる．本システムの構成の例を図 1 に示す．

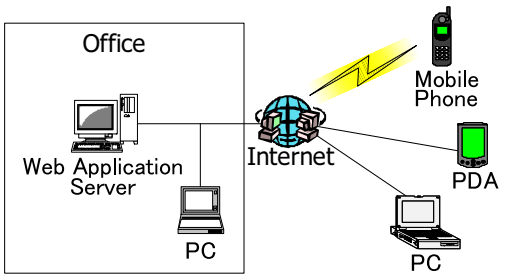

図 1: システム構成

作業引継ぎの従来の手法として，中断時の状態情報（状態の識別子等）を，サーバ/クライアント間でやり取りするセッション識別子に結びつけてサーバ側で保持しておき，再開時には保持していた状態と同一の状態に復帰する，という手法がある [2]．この手法は，単一の状態遷移図を全ての端末間で共有する場合には有効である．また，例えば，MDAT[3] では，異種端末を取り扱えるが，アプリケーションの振舞いが単一の状態遷移図で定義されるために，状態を容易に特定できる．

本環境では，作業の再開時に，中断時と異なる種類の端末を用いる場合には，中断時の状態遷移図と再開時の状態遷移図が異なる．このため，従来手法を用いて中断時の状態情報を記憶していても，その状態と同一の状態が再開時の状態遷移図上に無いため，作業の継続ができなかった．中断時の状態情報を利用するために，状態遷移図間の状態を対応付ける方法もあるが，状態遷移図や状態数が増える毎に対応付けの数が増大するため，定義情報の作成が困難となる．

3 共通アクションを用いた状態のマッチング

3.1 共通アクション

作業の進行状況を示す情報として，状態遷移図間で共通に使用されるアクション（以下，共通アクション）に着目した．本構築環境では，アクションを共通処理部で定義し，各端末向けの状態遷移図で共用できる．同一アプリケーションの新規端末向けの状態遷移図を追加する際は，アクションを流用する．

共通アクションには，全ての状態遷移図で共通に使用されるアクション，及び，一部の状態遷移図間で共通に使用されるアクションがある．本手法では，両方とも利用する．

3.2 状態のマッチング

本稿では，共通アクションを用いて状態遷移図間の状態のマッチングを行う手法を提案する．本手法は，1) 状態遷移図間で，同じ共通アクションを実行する遷移同士を等価な遷移とみなし，その遷移を境界として，状態 (集合) の対応づけを行う，2) 作業中断後の再開時には，中断時の状態に対応づけられた等価な状態を再開時の状態遷移図上で特定し，その状態から作業を再開する，というものである．

アクションには，遷移時に実行される遷移アクションの他に，状態に入る際に実行される入状アクション，状態から抜ける際に実行される出状アクションがある．遷移アクション以外は，主に画面に依存した処理を行うことが多いため，今回は遷移アクションに着目している．

以下，図 2 を用いて具体例を示す．図 2 では 2

Ubiquitous User Interface Middleware (4) - Application Program Continuation by Shared Action -
† Nobutoshi Todoroki, Misayo Kitamura, Akira Ishihara, Advanced technology R&D Center, MITSUBISHI Electric Corporation.

つの状態遷移図 (PC 用，携帯電話用) が定義されている．両方とも同じ機能を果たすものの，携帯電話は描画領域が小さいために画面が分割されており，携帯電話用状態遷移図は状態数が多い．図では，actionA 及び actionC は両方の状態遷移図上にあるため，共通アクションである．上記マッチングアルゴリズムによると，状態 state1-1 と状態集合 (state2-1，state2-2) が等価である．また，状態 state1-2 と状態 state2-3 が等価である．

次に具体的な引継ぎの動作の例を示す．PC 用状態遷移図にて state1-1 で中断した場合，上記等価の関係から，携帯電話用状態遷移図の state2-1 から再開される．携帯電話用状態遷移図にて state2-1，あるいは，state2-2 で中断した場合，PC 用状態遷移図の state1-1 から再開される．

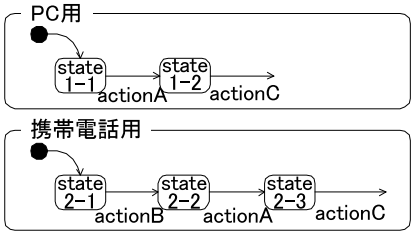

図 2: 共通アクションを含んだ状態遷移図

提案手法がうまく動作するための制約として，一つの状態遷移図上では共通アクションは一意に識別できなければならない．これは，再開時に再開状態を一意に特定するために必要である．

4　試作と提案手法の有効性の確認

提案手法を用いてシステムを試作した．本システムでは，アクションは ID で識別する．そして，状態遷移図間で同じ ID を持つアクションが存在すれば，それを共通アクションとした．

試作システムで用いた作業継続のアルゴリズムは次の通りである．作業中は，実行した遷移アクションの ID の履歴をサーバが保持する．再開時には，履歴の新しい方から順に ID を取り出して再開時の状態遷移図を検索する．検索が成功した場合に，そのアクションの実行後の状態を，再開時の開始状態とする．

このアルゴリズムによれば，全体に共通なアクションに加え，一部に共通なアクションも扱うことができる．このため，全体に共通なアクションの実行後に他のアクションの実行が続いても，全体に共通なアクションまで戻ることなく，より中断時に近い状態から再開することができる．

なお，アプリケーション内のロジックが使用する変数は，サーバが永続的に保持している．このため，作業を再開した際には，中断時の変数値をそのまま引継ぐことができる．

上記引継ぎ処理を実装した環境上で，事故・故障情報システムを試作した．本システムでは，発生した事故・故障の検索，及び，報告書の作成・閲覧ができる．状態遷移図の一部を図 3 に示す．図 3 には，PC 用，PDA 用，及び，携帯電話用の 3 つの状態遷移図が含まれている．PC 用は状態数 5 個，遷移数 11 個，PDA 用は状態数 12 個，遷移数 30 個，携帯電話用は状態数 17 個，遷移数 44 個である．また，全体に共通なアクションは 8 個 (図中では Login，Search，Show，Back)，2 端末間で共通なアクションは 5 個 (図中では，*State*, *Detail*) である．

上記試作システムにて，3 端末間では，状態遷移図間全てに共通な共通アクションに基づき引継ぎができた．また，2 端末間で共通なアクション *State*，*Detail* 実行後の状態にて中断した場合，当該 2 端末間では，全体に共通なアクション Show まで戻ることなく，同 2 端末間で共通なアクション実行後の状態から引継ぎができた．

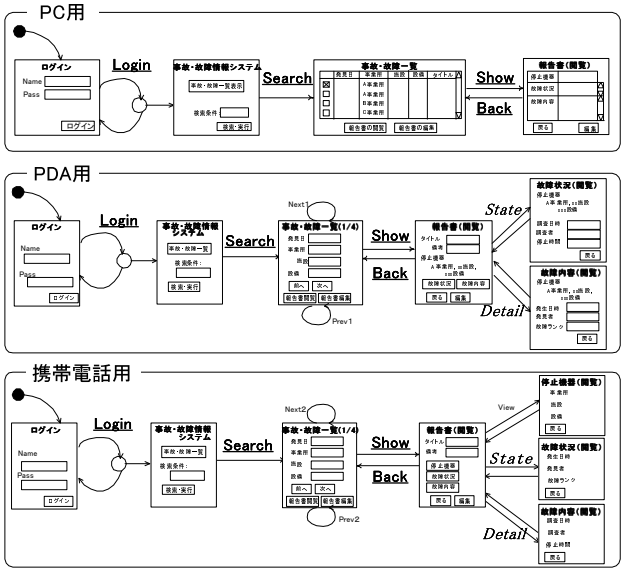

図 3: 状態遷移図（抜粋）

5　まとめ

本稿では，筆者らが開発するアプリケーション構築・実行環境における，異種端末間での作業の引継ぎ手法について述べた．試作システムにおいて，異種端末間で作業の引継ぎができ，提案手法が有効であることが確認できた．

参考文献

[1] 石原他: ユビキタスユーザインタフェースミドルウェア (1)-(3), 情報処理学会第 66 回全国大会 6G-1,2,3(2004).

[2] Danny Coward: Java Servlet Specification Version 2.3, http://jcp.org/en/jsr/detail?id=53 (2001).

[3] 北山他: MDA を用いたマルチデバイスアプリケーション開発, オブジェクト指向 2003 シンポジウム論文集 p.17-24 (2003).