研究論文

鳴門教育大学情報教育ジャーナル No.7 pp.15-22 2010

プログラミング講義における苦手箇所抽出法による学習支援環境の構築

皆月昭則*，林　秀彦**

本研究は，プログラミング教育における学生の苦手箇所を発見・抽出して学習支援する講義環境手法を構築した．実践したプログラミング教育の課題プロセスでは，学生が苦手箇所を考察し議論する創造的な活動で学習支援システムを開発完成し，次年度の学生で利活用することが可能である．

［**キーワード**：プログラミング，苦手箇所，C#，創造性，SECI モデル，組織的知識スパイラル］

1. はじめに

本研究では，学生の苦手箇所に着目（Anti Weak Point）したプログラミング学習支援システムを構築するために，苦手箇所を発見・抽出するための講義環境モデルを考案して実践した．図 1-1 が示すようにプログラミングの講義展開は，文法を理解しながら理解を深める．容易な文法の理解の支援として，講義資料が節ごとに「プログラミングの開発環境の解説，変数の取り扱い，制御文」などに分けられている．しかしながら，節の内容の理解に個人差があり，図 1-1 の②が示すような苦手な箇所の特定は教師あるいは他人による把握が困難である．また，小テストを実施する際の出題や作問では，これらの苦手箇所の特定が重要である．すなわち，学生自らの苦手箇所の抽出による作問が効果的な小テスト出題になるが，小テストの個別実施は困難である．

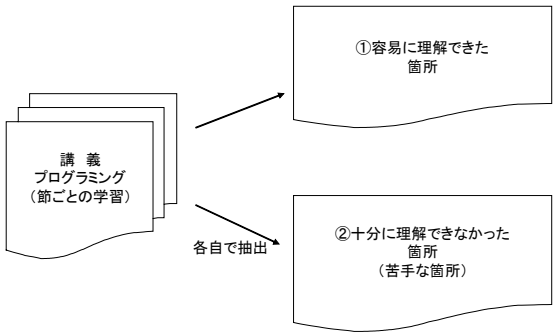

図1-1 苦手箇所を抽出した学習展開

よって，苦手箇所を抽出してから，どのように苦手箇所を克服するかが課題である．本研究では苦手箇所を学生個人からグループで解決する講義環境モデルの提案し，モデルの妥当性の検討において，組織的知識創造プロセスモデルで考察した．

2. 講義にける取り組みの考察

どのような科目であっても，「予習―講義―復習」が基本である[1]．一般的な予習と復習の方法論は，予習は講義を受ける前に該当講義の内容を学習しておき，理解できない箇所は講義を聴いて解決することである．そして，復習は，講義の後に振り返る学習である．特に復習に関しては講義の科目種別や内容にもよるが，ポイントとなる箇所や苦手箇所を発見・抽出しておくことが必要になる．また，復習的効果や理解度を確認するための小テストを実施することも有用である．しかしながら，小テスト実施における方法では，学生の予習・復習時間の一定の促進が期待されるが，結果（点数）を学生評価に反映させる際に注意が必要である．例として 2006-2007 年度の調査では，小テスト実施後にテスト結果を単位認定の際の評価に加算する場合に講義の辞退率に差が生じており，履修開始時の約 3 割の学生が講義期間の途中で辞退した．辞退理由を調査すると，学生はテストの獲得点数の状況によって，単位認定に影響が生じると推測すると，その時点で履修を辞退すると考えられる．この現象について次節でプログラミング教育に関して述べる．

2.1 小テスト方式の実施

プログラミング科目の「予習―講義―復習」過程で理解度や苦手箇所を特定することは容易ではない．そのような理由から，教師側から見た苦手箇所の抽出や理解度の確認のための小テストは有効であると考えられる．小テスト実施においては，学生が小テストで間違った箇所を再考することが必要である．本研究の講義実践では，15 回の講義の中で，8 回（TEST1～TEST8）の小テストを実施した．図 1-2 は，小テスト実施における採点返却プロセスの終端行動を示しており，採点返却後の条件分岐の復習行動が間違い箇所の確認，苦手箇所の特定にも結

*　釧路公立大学　情報センター
**　鳴門教育大学 大学院 自然・生活系教育部

びつく場合がある．苦手意識の検討やペア学習などは，先行研究[2]で報告されているが，本研究の講義実践における小テスト実施採点返却後の復習状況調査の考察を次節で述べる．

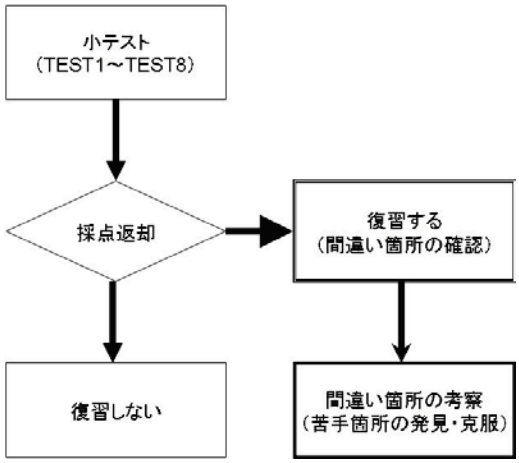

図 1-2 小テストの効果とその後の条件分岐行動

2.2 小テスト実施方法と復習調査の考察

履修辞退率を低下させない方策としては，試行的に小テスト結果を単位認定の評価に加算しないようにすると，履修辞退率が改善し，辞退学生が約 1 割になった．しかしながら，この方策で復習を促進させることは困難であることが調査から明らかになった．図 2 に示した復習割合の定量的な示唆以外に，TEST8 後に実施した復習に対する学生の意識調査で定性的な特徴が明らかになった．A 学生の場合，履修開始時から数回の講義は，簡単な講義内容で興味関心もあったが，TEST3 時期以後から難しくなり，その後は，小テストの結果を振り返らなくになった．B 学生の場合，小テストが単位認定の評価に加算されない安心感で，予習も復習もしないなど他，複数回答が得られた．

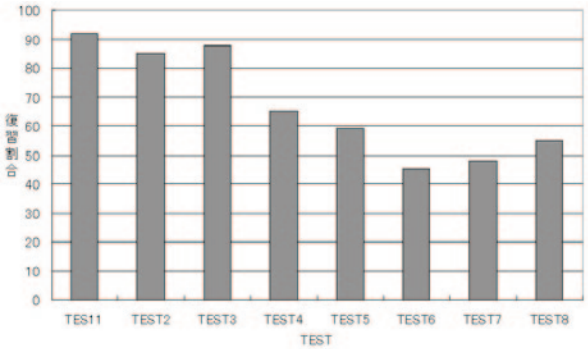

図 2 小テスト返却後の復習割合

プログラミング講義は，言語の種別に関係なく内容は，ほぼ類似した展開になると考えられる．本研究のために実践した講義では，C#言語を用いている．シラバス的な内容は，①開発環境支援ツールの基本操作，②プログラミングの基本（オブジェクトとコントロール，イベントメソッド，プロパティ，変数と定数，演算子，座標），③制御文（条件判断・複数方向分岐，所定回反復・前後判定反復，多重ループ，強制脱出），④データ型と配列および構造体（データ型・サイズ，ローカル変数・ファイル変数，一次元配列，二次元配列），⑤クラスライブラリ，⑥ユーザ定義メソッドと多重定義，⑦ファイル処理･ディレクトリー操作である．図 2 が示す TEST3～TEST4 以降の復習率低下時の内容は，ネストや構造化プログラミング関連の箇所で，TEST2 頃までの個人対処では，苦手箇所の発見や克服のための学習（復習）対応ができなかったと考えられる．これは，先行研究[3A][3B]で指摘されている傾向に類似しており，制御文の学習内容や時期から苦手意識が高まることが推察できる．

2.3 苦手箇所を記述するカードの考案

2006-2007 年度の講義実施後の検討考察の結果，2008 年度は小テストを中止した．これに代わる手法として，講義開始時に配布したカードに毎講義後に苦手箇所を自由記述させた．この苦手箇所抽出カード（Weak Point Card；以下 WPC）は，図 3 に示すカード 2 枚で，毎講義の終わり約 5 分間で 300 字以内（あるいはカードの罫線枠内）の苦手コメントを記述して各自で保管させるようにした．

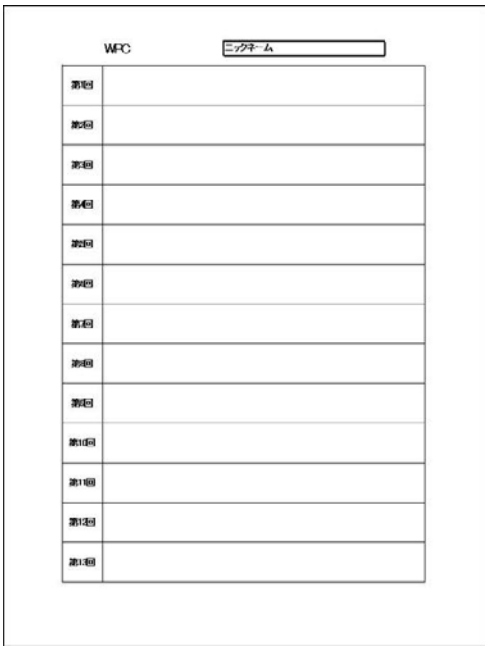

図 3-1　苦手箇所抽出カード（WPC）

3. 苦手箇所の克服への開発課題

苦手箇所の発見・記録手法では，前節で述べた WPC を使用することで，毎講義毎に各自の苦手箇所が復習あるいは再考必要箇所になる．しかし，WPC は，学生が講義後に実行的に対処にしなければ，苦手箇所の克服ができ

ないと考えられる.

3.1 WPC 情報にもとづく開発課題決定

WPC は，毎講義後に，限られた文字数で，苦手項目を列挙しておくことである．列挙例としては，C 君；サンプルコードや練習問題でローカル変数の指定箇所がわからない，D 君；リスト内容を条件分岐で処理できないなどが書かれている．

本研究では WPC を 2 段階で活用する方法を考案して講義で実践した．第 1 段階では，講義の中盤（第 5 回または第 6 回）で，WPC を回収して，無作為に 4 枚（名）づつ選んだ学生グループを作り，他者と WPC を照合比較しミニ勉強会の時間を設定して苦手箇所克服の成果を発表させた．

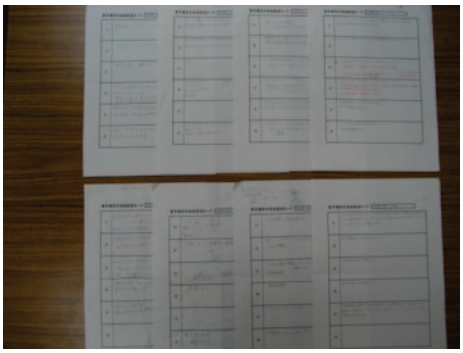

図 3-2　4 人の WPC 抽出（上；前半・下；後半）

第 2 段階では，講義の終盤（第 11 回または第 12 回）で，単位認定課題として学習支援システムを開発するグループを作る．このグループメンバー構成は 4 名で，まず，教師が WPC から無作為に抽選（抽選数＝WPC の全枚数/4）する．そして，教師に抽選された学生が，残りの WPC を閲覧してグループを構成する 3 名を選出する．その後の選出では，二人目・三人目のメンバーも WPC を閲覧して選出合議を通じて四人目のメンバー候補が決まる．WPC には，苦手箇所とニックネーム（1 枚目の WPC と異なる名称）のみが記述されており，選出合議で個人を特定することが困難である．

グループが決定されると,図4に示したようにメンバー 4 人の WPC を照合して苦手箇所を抽出して，それらを克服するための学習支援方策を企画検討する．検討ツールの ASP は，苦手克服システムの名称（タイトル）決定，フローチャート，開発システムイメージを ASP のデザイナー機能で作成し，グループで発表する．発表で指摘された企画を修正後，約 4 週間でプログラミング苦手箇所を克服するための学習支援システムを単位認定の課題として開発した．

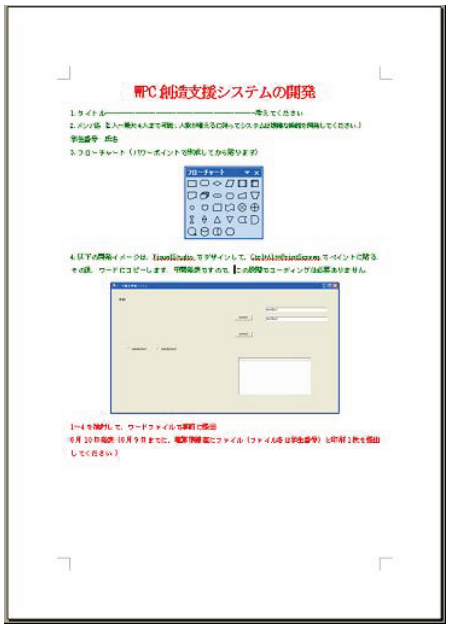

図 4　WPC 学習支援システムのグループ企画

3.2 WPC 情報から CSS-AWPS の開発

WPC 情報は，各自の苦手箇所を記述して保管するだけでなく，苦手箇所を，どのように克服して理解学習できるのかを討議に活用できる．すなわち，各自の WPC で，どのような箇所が苦手で，克服のためにどこまでの学習範囲をフォーカスするのかを，グループで導出する．そして，学習支援するための仕組みや手法を開発する．苦手箇所を克服（Anti Weak Point System；AWPS），すなわち，苦手箇所に気づき，つまずかないための学習支援方策を創造的（Creativity Support System; CSS）に企画開発するためのツールとして CSS on the AWPS と名付けた．本システムは苦手箇所を保有・共有する学生グループで開発されており真の苦手箇所や意識を考慮している．開発過程の実践は苦手箇所発見・抽出と創造的企画開発プロセスの融合で図 5 に示すモデルになる．

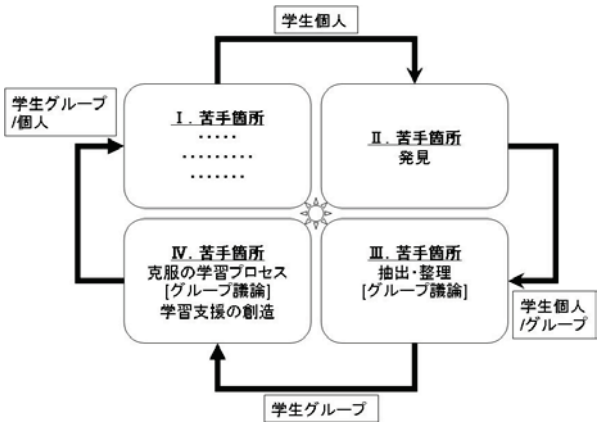

図 5　苦手箇所情報から知識創造（知識変換）

学生個人の苦手箇所は，図 5 に示すようにⅠ，Ⅱ，Ⅲ，

Ⅳのように個人の対処からグループの対処になる．図 5 の流れでは,WPC 情報に記述された苦手箇所情報がグループで議論される．議論のプロセスでは，互いに教え学びあう関係が，しだいに成立していることも観察で明らかになった．教え学び合う過程では，WPC で記述されている苦手箇所がその議論で学習理解される場合もある.よって，議論で本来（真）の苦手箇所を発見・抽出「表出化」し，また，それらを「連結化」して，学生意識の苦手箇所克服のための学習支援ツール開発のフォーカス範囲の操作知を導出した.

図 5 におけるモデルは，本研究における講義実践によるが,組織的知識創造の理論[4]に類似しており比較検討した．組織的知識創造の理論では，図 5 における上段から下段の移行時期を理論[4]と比較すると[対話/行動による学習]時期として「知識スパイラルとその内容の変化」時期として WPC で苦手箇所を表出化した概念知であり，さらにグループで連結化した体系知による苦手箇所を克服するための学習支援プロセスを開発する操作知の導出になる．したがって，小テストを実施しない方法で，苦手箇所の克服のための知識変換・獲得という実践的モデルが履修率向上と学習に関わる「復習」機会や意欲を向上させたと考えられる.

3.3 苦手箇所共有による CSS-AWPS の開発例

［3 年・4 年向け講義（プログラミング論Ⅱ）］

図 6-1-1 および図 6-1-2 と図 6-2 は WPC 情報と図 5 で示したモデルを背景に開発した学習支援システムの起動画面のスクリーンプリントである.

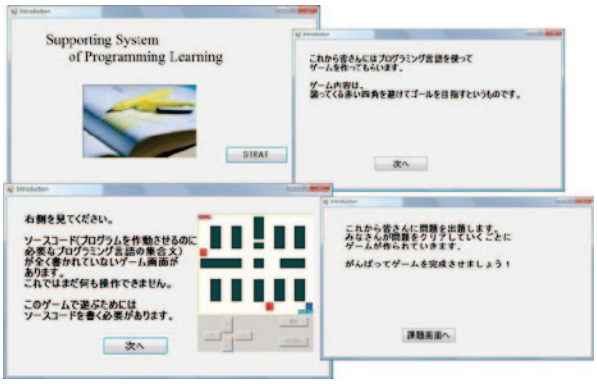

図 6-1-1　日本人学生の AWPS（起動画面）

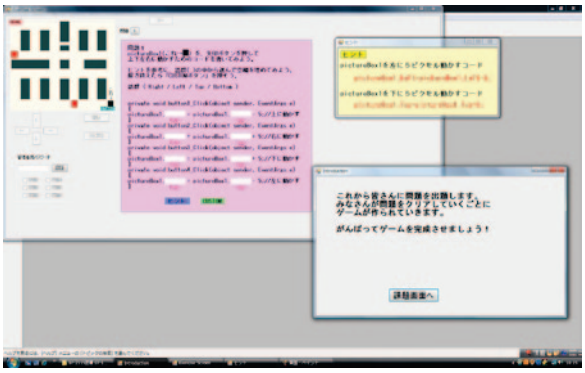

図 6-1-2　日本人学生の AWPS（制御文関連）

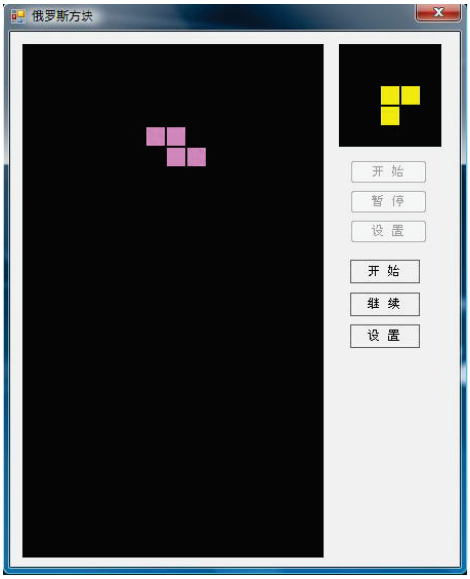

図 6-2　留学生（台湾）の AWPS（配列関連）

3.4 苦手箇所共有による CSS-AWPS の開発例

［2 年向け講義（プログラミング論Ⅰ）］

図 6-3～図 6-6-4 までは,学生によるシステムである．図 6-3 に示すように，初期ではシンプルな学習支援であるが，1 ヶ月程度の開発期間で，画面遷移などの機能が付加されるなど図 6-4 から示したシステムになっている.

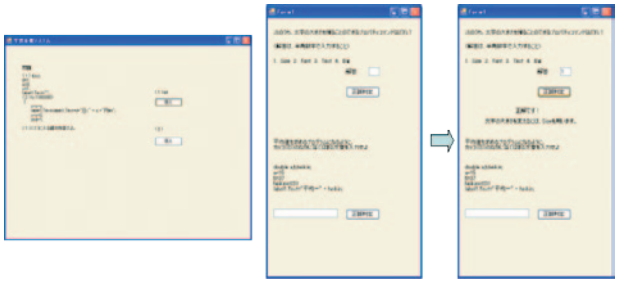

図 6-3　初期のシステム（2 年生）

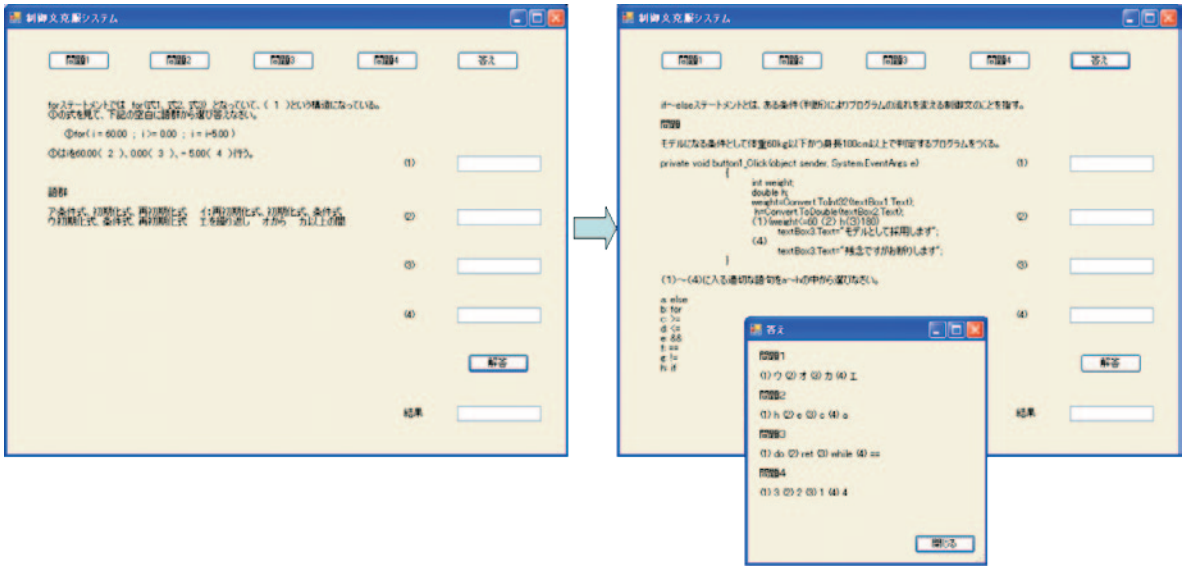

図 6-4　グループ A

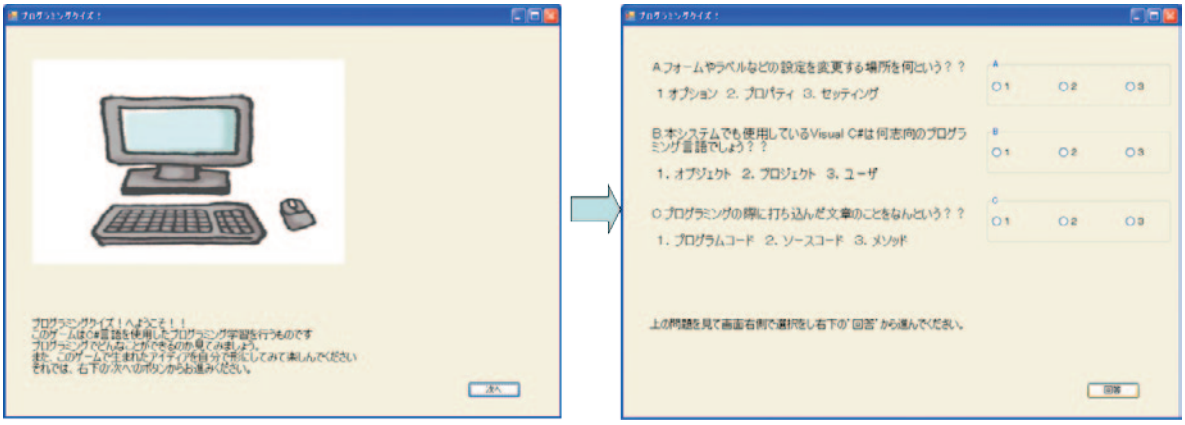

図 6-4-1　グループ B（メイン画面）

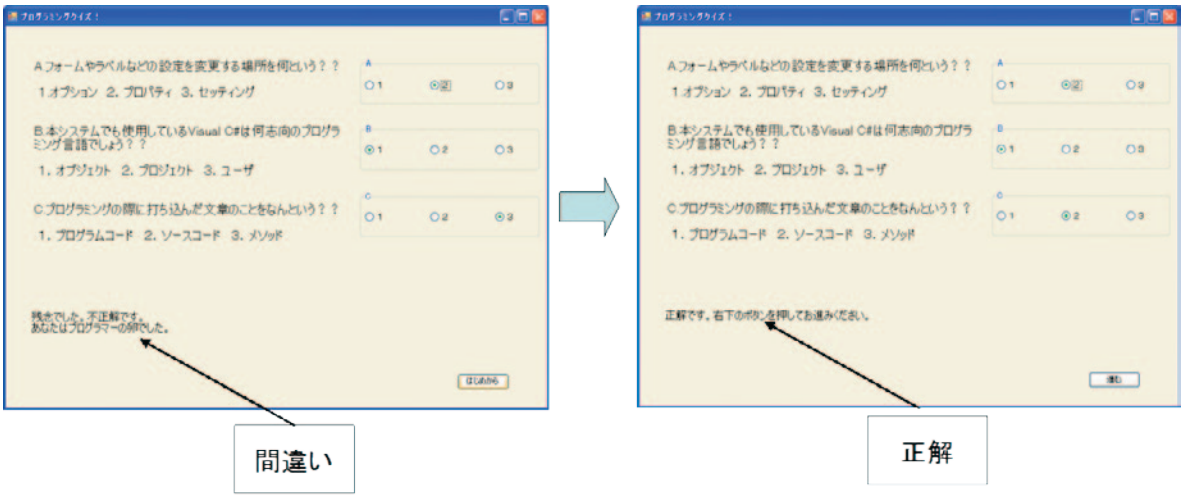

図 6-4-2　グループ B（画面遷移・左；不正解・右；正解）

図 6-4-3　グループ B（正解の場合の画面遷移）

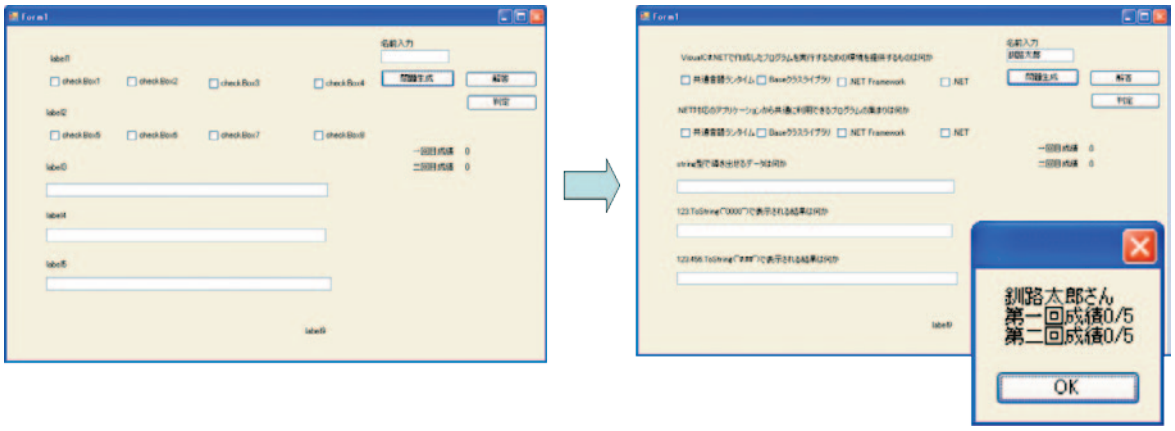

図 6-4　グループ C

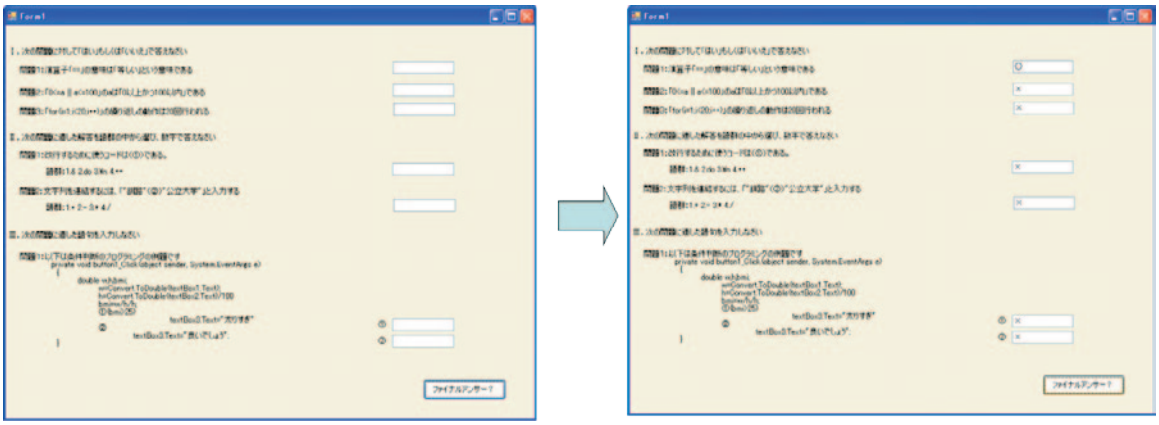

図 6-5　グループ D

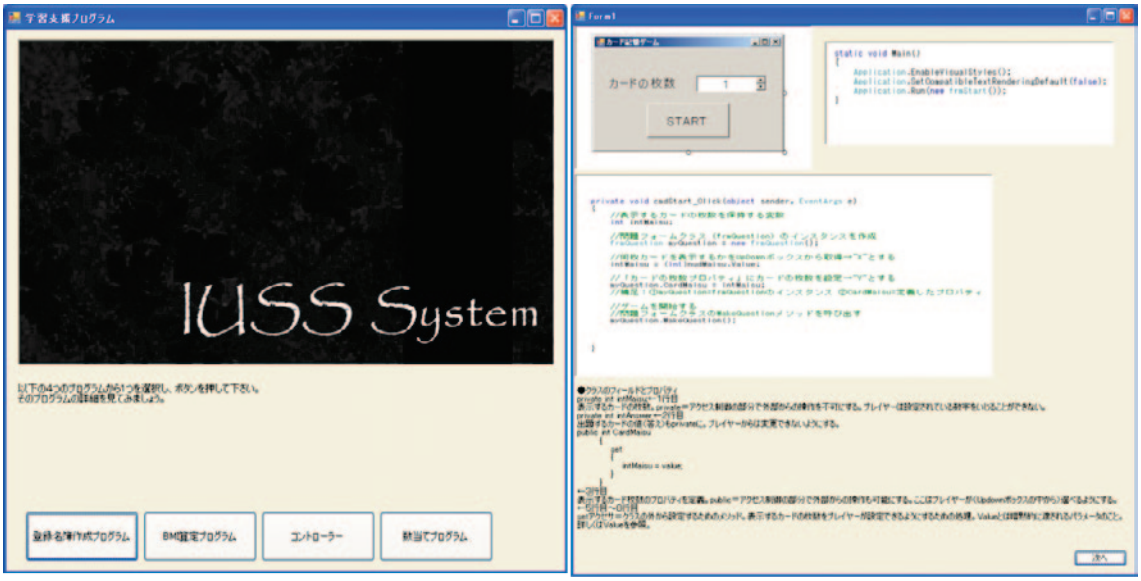

図 6-6-1　グループＦ（メイン画面）

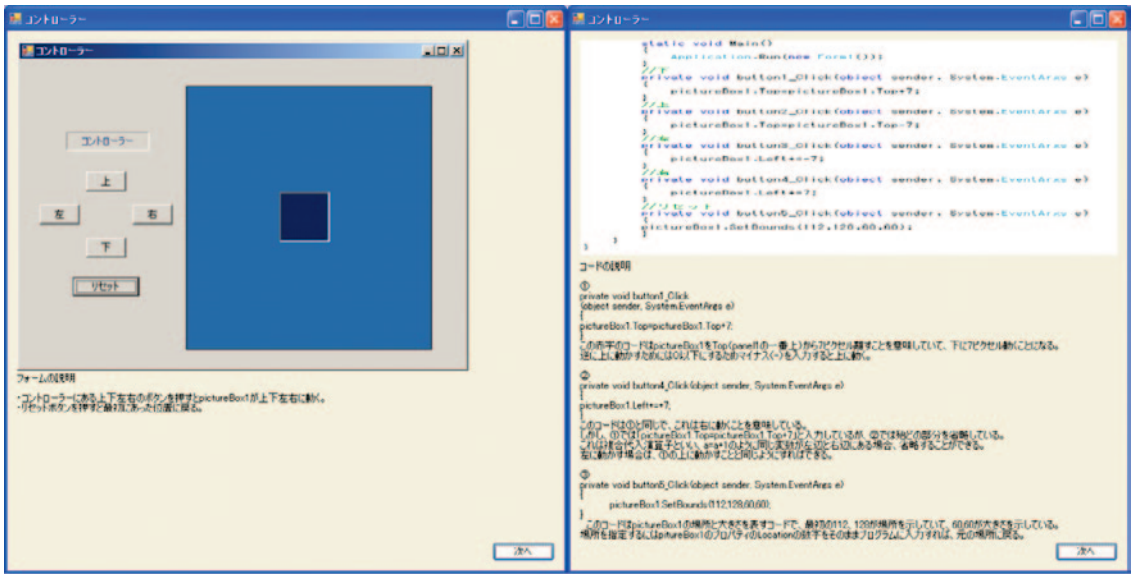

図 6-6-2　グループＦ（画面遷移）

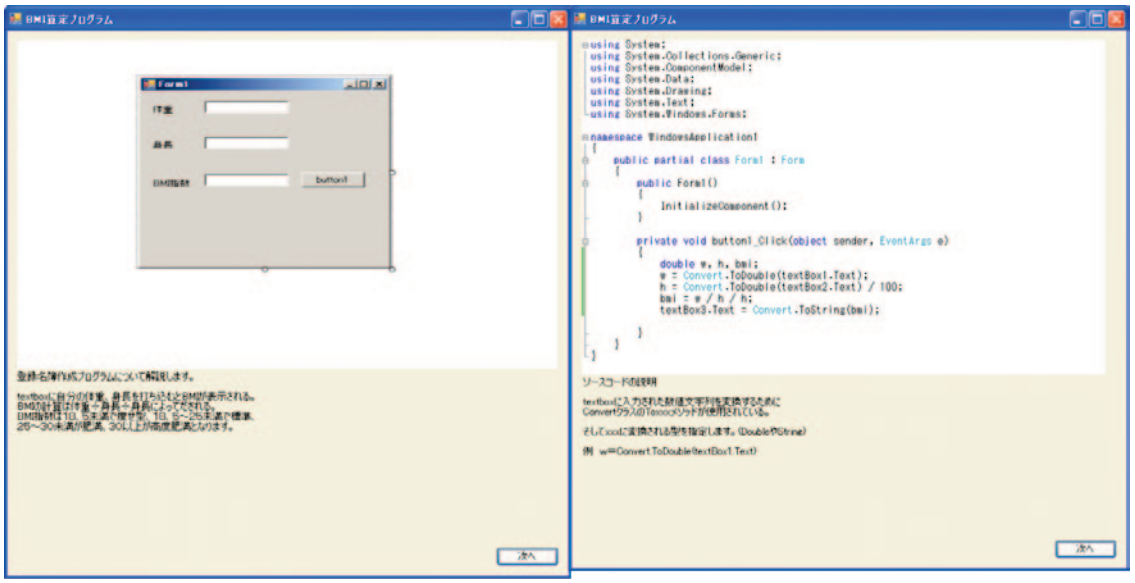

図 6-6-3　グループＦ（画面遷移）

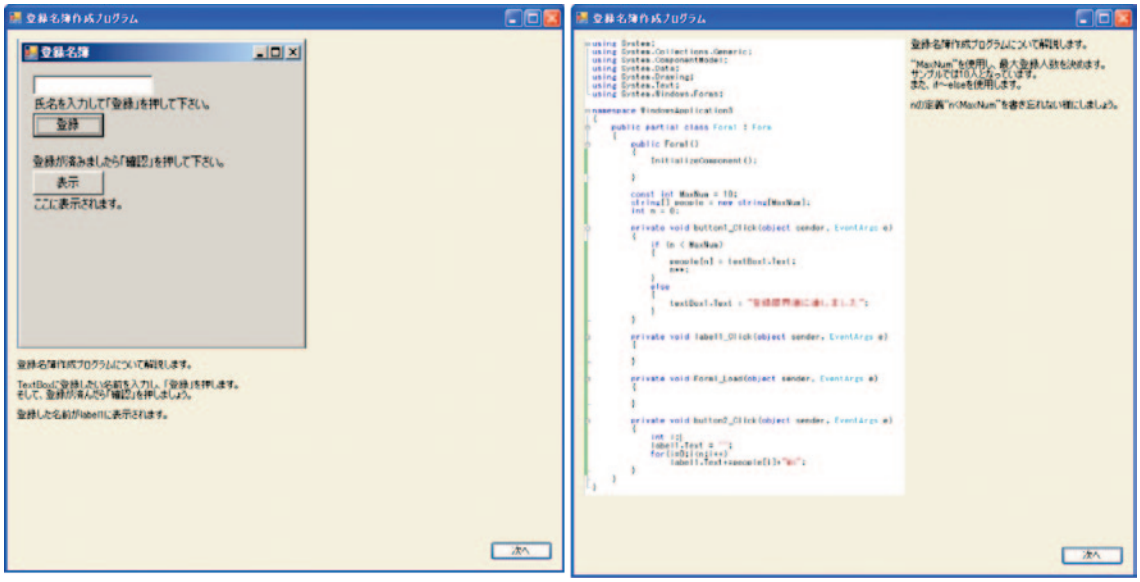

図 6-6-4　グループ F（画面遷移）

4. AWPS の利活用(feedforward-Learning)

フィードバックは学習結果を受けて調節する振り返り型であるが.本提案では目標を先に決めて苦手箇所・要因を評価しつつ，プログラミングの言語学習達成に向けた修正を加えるフィードフォワード型の学習制御ツールを考案開発した．苦手箇所があらかじめ提示解決支援する創造的ビジョン優先のフィードフォワード的な学習支援システムとして図 7 にインターフェースを示す．本システムは，学生グループが開発した複数の AWPS が次年度以降の学生にも利活用できるようにしており 2009 年度からの講義学習支援に活用している．

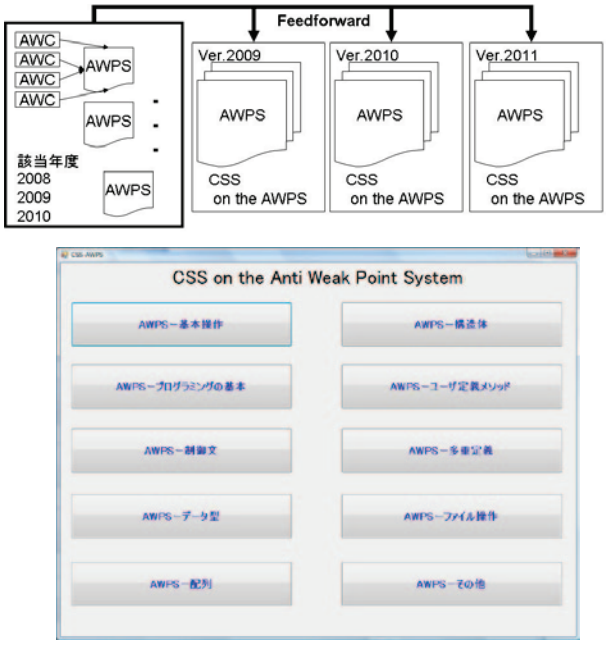

図 7　次年度の利活用の feedforward-Learning

5. おわりに

一般的に教育とは，教師による教授法や資料が異なり，オリジナリティが付随する．よって，E-leaning も教師のオリジナリティなど講義の特質が含むことで学生が親しみやすくなる．本研究では，著者（教師）の特質である教授法や資料の中から学生が苦手箇所を抽出して，オリジナルの leaning システムを実装構築しており，実装側の該当学生と次年度にシステムを使用する学生の苦手視点や教授法の癖などの視点を包括的に取り入れたプログラミング教育環境を提案した．

参考文献

[1] 名古屋大学高等教育研究センター編，「ティップス先生からの 7 つの提案」（学生編，教員編，大学編），2005

[2] 崎山 充，「プログラミング教育における導入期の苦手意識の変化に関する一考察」，FIT2009（情報科学技術フォーラム）講演論文集，2009

[3A] 生田目康子，「ピア・レビューをともなうグループ学習の評価」，情報処理学会，2004

[3B] 田口浩，「個々の学習者の理解状況と学習意欲に合わせたプログラミング教育支援」，情報処理学会，2007

[4] 野中郁次郎，「知識創造企業」，東洋経済新報社，1996

[5] 藤岡　直矢，「C＃言語学習における「発想 - 創造」過程を含むエデュテイメント学習コンテンツの開発」，PC カンファレンス CIEC2009，2009