情報システム分析・設計論の学習を動機付ける プロジェクトシミュレータの開発と検証

花 川 典 子

A Project Simulator for Discovery Learning of Software Engineering

Noriko HANAKAWA

和文抄録：

Webベースの統合教育支援システムHInT（Hannan Internet communities Tool for E-Education）を利用してプロジェクトシミュレータによる発見学習を含むソフトウェア工学教育を実施した。まず、HInTは学生の大学生活を事務システム、教育システムの分断なくサポートするシステムである。特徴はいつでもどこでも学習できる環境を提供したところである。このHInT上のコミュニティを利用してプロジェクトシミュレータによる発見学習をソフトウェア工学の予習として実施した。通常数週間単位の長期間を要する発見学習であるが、シミュレータで模擬体験することにより７時間から10時間で発見学習が実現できた。６名の被験者学生のうち５名が発見学習で知識獲得でき、さらにレギュラー授業の最終テストでも良い成績を修めた。これによってシミュレータによる発見学習がソフトウェア工学教育において有効であることが確認できた。さらに、HInT上の教育サイクルと学生カルテ、教材DBを利用することで、学生の個人別能力に応じた個別学習プログラムの自動生成が可能であることが確認された。

キーワード：

Webベースシステム、教育支援、発見学習、プロジェクトシミュレータ

欧文抄録：

A web based integrated education system: HInT（Hannan Internet communities Tool for E-Education）has been developed in Hannan University. HInT consists of portal sites for individuals, e-learning sites, and connection facilities to administration information. HInT supports sufficient communication between teachers and students. Using HInT, we tried to do discovery learning for software engineering. The purpose of the discovery learning is that students understand complicated phenomena in software development projects. While the students execute a project simulator on the e-learning sites, the students acquire/discover

knowledge about the complicated phenomena of the projects. As a result, the students were able to understand the complicated phenomena without real execution of software development projects. In addition, we confirmed that the students who did discovery learning remembered longer time than the students who took lectures about the complicated phenomena.

1．はじめに

現在の日本の大学教育の問題点は18歳人口の減少である。日本は高齢社会へ移りつつあり、若年層の人口比率は下がる傾向にある。1995年の18歳人口が175万人であったのに対し、2008年の18歳人口は121万人になると予測され、13年間で30%の減少が予想される。この18歳人口の減少は日本の大学教育に対して深刻な影響をあたえている。最も大きな影響のひとつは学生の学力の低下である。18歳人口が減少しているにもかかわらず日本の大学全体のキャパシティが減少したわけではないので、大学教育にふさわしくない低学力の学生であっても大学に入学可能である。特に、私立大学は経営的観点から考えると、低学力の学生であっても受け入れざるおえない状況にある。結果として、教員は十分な学力を持つ学生に対する教育だけではなく、不十分な学力の学生にも教育サービスを提供する必然性が生じる。様々な学力をもつ学生へ教育サービスを提供しなくてはならないので、教員の仕事も多様に広がっていく。教員は授業やゼミにおいて、一般的な大学の教育レベルを維持しつつ、不十分な学力をもつ学生にも理解可能な講義を工夫する必要がある。教員はもはや単純な講義形式の授業実施だけでは学生の要求を満たせない。このような状況において学生への学習動機付けが教育の中の重要な要素となる。特にe-learningやインタラクション教育、webベースの教育などの様々な教育手段を使って学生の学習モチベーションの維持が必要となる。更に、様々な学力をもつ学生たちには様々な学習機会を提供する必要が生じる。例えば、授業時間帯だけでなく、自宅や通学途中の電車の中など、いつでもどこでも学習できる環境を提供することも重要な要素となる。

そこで我々はwebベースの統合教育支援システムHInT（Hannan Internet communities Tool for E-Education）を構築し、その上で様々な学力の学生に対応するソフトウェア工学教育を試みた。webベースの統合教育支援システムHInTとは、個人別ポータルサイト、すべての授業のe-learningサイトなどから構成され、事務システムとのデータ連携も実現している。さらに教育サービスのみならず就職部や学生部からの事務サービスも提供でき、学生の大学生活を統合的にサポートするシステムである。HInTに関する詳細は文献[4]に記載されている。どこでもいつでも学習できる環境のHInTを利用してソフトウェア工学に関する教育を行った。ソフトウェア工学教育は学生の動機付けが難しい授業である。なぜならば、学生は実習程度の小規模プログラムしか作成した経験がなく、大規模システム開発で要求されるソフトウェア工学の必然性

を実感しにくいからである。そこで、本教育では、毎週の講義形式の知識伝達型授業の前に、システム開発を模擬実行できるプロジェクトシミュレータを使った予習プログラムを実施した。本シミュレータの特徴は実際のシステム開発プロジェクトのエッセンスを模擬実行できるところである。単純な課題を事前に与え、シミュレーションを繰り返すことによって、学生の問題意識や動機付けを行うことを予習とした。これによって講義形式授業を効果的に行うことができた。

本稿では、2章で関連研究を示し、3章でHInTの簡単な紹介を行う。4章ではHInT上で実施したソフトウェア工学教育の効果について報告し、その考察を5章で行う。6章にてまとめと今後の課題を述べる。

2．関連研究

ソフトウェア工学教育に関して多くのシミュレータやWebベース教育システムが提案されている。まず、コンピュータを使った教育システムの紹介を行う。同期、非同期のe-learningサービスを提供する仮想協調空間：EVE（Education Virtual Environment）がBourasらによって開発された [2]。また、コンピュータサイエンス教育のためのクライアントサーバシステムで制御された教育システムも構築された[9]。このシステムの特徴は著者らのオリジナルなデジタルモデルレールロードプラットフォーム、つまりコミュニケーションスキルを工学カリキュラムへ組み込むために、ワークスペースを統合したwebベースシステムである。これによって、効果的な教育の実施と共に教員と学生の十分なコミュニケーションを確立することができた。本システムによって様々な学生への多様な教育サービスが可能となった。HInTシステムもwebベースシステムのひとつである。ただし、HInTの重要な特徴は教育システムの範囲を超えた統合教育支援システムであることだ。つまり、HInT上で実施される教育は特定教育活動のみならず、大学の活動全体と事務サービスも含むことができるということである。幅広く対応したHInT上での教育は単なる教育の枠を超えた大学全体のサービスを向上させることができる。

次にソフトウェア工学教育について述べる。ソフトウェア工学では、コンピュータによるシミュレーション教育が注目されている。 Drappa[3], Blake[1], や Oh, E[8]らはソフトウェア工学の学習の難しさを認識し、シミュレーション手法にてソフトウェア工学を教授した。学生はシミュレーションにて仮想発見学習や仮想体験学習を実施することができる。実体験の発見学習や体験学習は効果的ではあるが、非常に長期間を必要とする。しかし、シミュレーションを使えば発見学習も体験学習も短時間で完了することが可能である。我々が行ったシミュレーションによる教育も同様の考え方に基づいている。ただし、シミュレーション教育においてはシミュレーションを実行するモデルが重要なキーポイントとなる。シミュレーションモデルが講義目的を的確に計算し表現できるか、そのモデルは十分に評価されているかが問題となる。

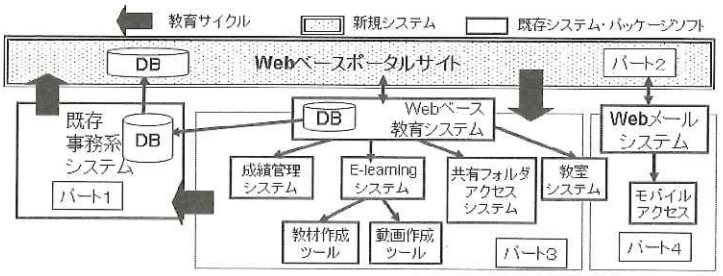

図１：HInTの概要

我々のモデルも様々なモデルを実装することが可能であるが、従来のシミュレーション教育と同様に講義目的との的確な連携を達成する必要がある。

3．統合教育支援システム：HInT

まず、最初にシミュレーションを用いたソフトウェア工学教育実施のインフラであるHInTを簡単に紹介する。本システムは大きく４つのパートに分かれる。既存の事務系システム、ポータルサイト、教育システム、さらにWebメールの４パートである(図１参照)。本システムの特徴は、既存技術を融合して大学教育における有益な教育サイクルをシステム上に完成させたことである（図１の太い矢印参照)。教育サイクルとは事務系システム（パート１）を開始点にして、ポータルサイト（パート２）での事務と教育情報の相互参照、さらに教育システム（パート３）へ進んで教員と学生による授業実施や予習復習、テスト等を行う。学習後に授業における出席点、課題点、テスト点を教育システム中で半自動採点し、教員による最終成績決定を経て、最終的には開始点である事務系システム（パート１）の成績データを更新するという事務系と教育系のシームレスな教育サイクルである。

本教育サイクルをHInT上に完成したことによって、阪南大学はオリジナルな教育を実施することが可能となった。例えば学生の教育カルテ（図２参照）である。過去の教育履歴とその成績を参照することによって、学生の様々な学力や能力を教員が共有できる。その教育カルテに基づき、これから行われる教育の前提知識の有無を確認することができ、必要であるならば個別の予習プログラム、復習プログラムを準備することが可能となった。

4．ソフトウェア工学の発見学習

4．1　ソフトウェア工学教育の問題点

ソフトウェア工学教育の大きな問題点のひとつは動機付けの難しさである。ソフトウェア工学はソフト開発における非常に重要な知識であるにもかかわらず、大きなプログラムを複数人

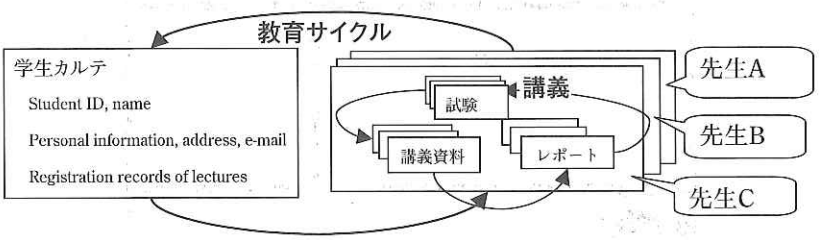

図2：学生カルテを利用した教育サイクル

数で開発した経験のない学生にとってその必然性を理解することが困難である。つまり、学生ひとりで小規模なプログラムを作成する場合、特に設計やドキュメントを充実させる必要がない。もし、作成したプログラムに関して不都合があれば、もう一度すべて作り直せば済むことである。一人で十分に小さいプログラムを作成している場合にのみ、有効な解決策である。大学教育におけるプログラム作成実習はこの範囲である。したがって、要求仕様の分析やソフトウェアの設計などプログラミング以前の作業の必然性を理解する機会がない。もちろん、ソフトウェア工学の授業の最初で分析や設計の必然性を講義するが、一方通行的な知識伝達型講義なので分析や設計の必然性を実感することができない。したがって、ソフトウェア工学を学習した後も、学生はいきなりプログラミングからソフトウェア開発を開始することがしばしば発生する。その後、プログラムが仕様を満たせないとわかると簡単にすべてを破棄し、新しく作り変えることを繰り返している。

ソフトウェア工学の重要性を十分に理解し、効果的な教育を実施するためにもソフトウェア工学学習への動機付けが非常に重要な要素となる。

4．2　シミュレーションを使ったソフトウェア工学の発見学習

ソフトウェア工学教育をさらに効果的に実施するために、HInT上にプロジェクトシミュレータを構築し、発見学習が可能な環境を提供した。これをソフトウェア工学講義前に予習プログラムとして組み込むことにより、学生の動機付けを行った。本発見学習の概要は以下のとおりである。

- ソフトウェア開発プロジェクトシミュレータはHInTのe-learningサイト上に構築される。
- シミュレータはソフトウェア工学の重要な概念に基づく複雑な現象を含む様々な振る舞いを模擬実行する。
- 学生はシミュレータの様々なパラメータを変更することでプロジェクトの様々な振る舞いを発生させ、同時にプロジェクトで発生する現象の規則性を発見する。
- シミュレータで試行した後、学生はシミュレータに組み込まれたソフトウェア工学の要素技術をそれぞれ発見する。

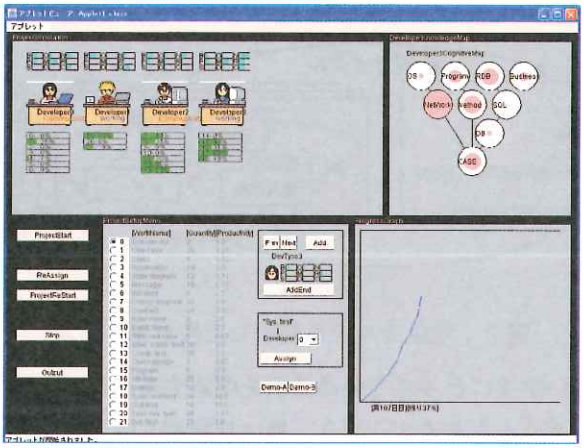

図３：プロジェクトシミュレータの画面イメージ

教員は、学習してほしいソフトウェア工学の重要な概念を直接学生へ教えることはない。学生は多くのシミュレーション実行結果を通して、重要な概念を学生自身で発見する学習方法である。

４．３　発見学習の計画

まず、我々はJAVAにてソフトウェア開発プロジェクトシミュレータを開発した。本プロジェクトシミュレータは現在、重要な２つのコンセプトに基づく現象を発生させることができる。ひとつめのコンセプトは開発者の知識と作業実行に要求される知識の関係によって生産性が変化するという現象である[5][7]。つまり、作業を実行するには特定の知識が必要とされ、同時に開発者も固有の知識を持っている。両者の相互的な関係によって作業実行の生産性が決定されるという現象である。さらに、開発者の知識は作業実行に伴って増加、すなわち学習するという振る舞いも計算できる。もうひとつのコンセプトは作業分割による開発者間のコミュニケーションの発生である [6]。つまり、関係の深い作業を別の開発者に分担すると開発者たちは多くの情報伝達、コミュニケーションをしなくてはならない。反対に独立性の高い作業をそれぞれ開発者へ割り当てると、開発者間コミュニケーションはあまり発生しない。それぞれの現象はソフトウェア工学分野にて研究されたソフトウェア開発モデルであり生産性を計算できる仕組みになっている。図３に本プロジェクトシミュレータ画面イメージを示す。学生はプロジェクト管理者の役割を担い、固有知識を持つ開発者へ作業割り当てを行う。作業はオブジェクト指向開発方法論で定義された22種類の作業であり、作業実行に要求される知識と作業間の関連の強さをあらかじめ定義されている。

本シミュレータを使った発見学習において教員は「22種類のすべての作業を100日以内に終了させてください」と非常に単純な課題を与える。当然、教員は前述の２つのコンセプトについてはなにも説明しない。シミュレータを使った発見学習の中でこの２つのコンセプトを獲得

することを目的とする。また、100日以内で残作業を終了させることは、2つのコンセプトを十分に理解しなくては達成できない目標であり、100日以内に終了させると自動的に学生が2つのコンセプトを獲得できているという仕組みである。

本発見学習で獲得してほしいコンセプトの具体的な項目は以下のとおりである。

C1：オブジェクト指向開発方法論と作業の関係

C1-1：総合テストフェーズの作業と分析フェーズの作業には強い関連がある。

C1-2：プログラミング作業と単体テスト作業を異なる開発者に割り当ててはいけない。

C1-3：量の多い仕事は、作業分割よりは、十分な知識の一人の開発者へ割り当てるほうが良い。

C1-4：クラス図作成と結合テスト作業を分担してはいけない。

C2：開発者の知識と作業実行に要求される知識との関係

C2-1：知識量の多い開発者に作業を割り当てると生産性は高いが、開発者の知識が増加することはない。

C2-2：知識量の少ない開発者に作業を割り当てると生産性は低いが、開発者の知識は作業実行に伴って増加する。

ソフトウェア工学の発見学習のステップを以下に示す。

S1：授業の受講者より数名の学生を選択する。

S2：HInT上での発見学習実施

S2-1：学生はシミュレータの使い方を学習する。数回試行する。

S2-2：課題である「100日以内に全作業を終了させる」をシミュレーションで達成するために、様々な作業割り当てを試す。課題が達成できるまで何度でも繰り返す。

S2-3：100日以内に全作業が終了できた後、あらためて学生に最も良いと思われる作業割り当てを行いシミュレーションさせる。さらに、教員は学生にシミュレーションで何を学んだかを質問する。

S3：教員は通常のレギュラー授業で全学生（発見学習した学生を含む）に対して、前述のC1-1からC2-2の内容を講義する。

S4：数ヵ月後、最終試験で前述のC1-1からC2-2の質問を行い、発見学習をしたグループとしないグループを比較する。

4．4　発見学習の結果

6名の学生をレギュラー授業より選択し、HInT上でシミュレータによる発見学習を実行した。上記のS2-2のステップでは、課題の「100日以内に作業実行終了」の達成が非常に困難であったため、結果として7時間から10時間の長時間を必要とした。発見学習の結果を表1に示す。学習項目の列が4.3の発見学習項目C1-1からC2-2の6項目に相当する。S2-1とS2-3の列は、前述のステップS2-1とステップS2-3の学生の知識量を意味する。“○”は学習項目C1-1からC2-2の知

識が十分にあることを示し、“△”は一部の知識、“×”はその学習項目の知識が全くないことを示す。すなわち、ステップS2-1で“×”であった学生はステップS2-3で“○”になっていると発見学習において非常に知識を獲得できたことを示す。反対にステップS2-1で“×”であった学生はステップS2-3でも“×”であると、発見学習でほとんど知識獲得できなかったことを示す。

発見学習での知識獲得は、プロジェクトシミュレータ内での開発者の作業割り当てのログで計測した。つまり、学生へのインタビューによって知識獲得状況を計測するのではなく、ステップS2-3の作業割り当てログの客観的データに基づき計測することを基本とした。例えば、学生はC1-1の学習項目の知識を獲得したとすると、シミュレーション上で同じ開発者が分析フェーズと総合テストフェーズに割り当てられたはずである。すべての学習項目に関して、ステップS2-1の作業割り当て方法とステップS2-3の作業割り当て方法を比較した。もし、学生がS2-1で悪い作業割り当てをしていたにもかかわらず、S2-3でよい作業割り当てになっていたら、その学生はステップS2-2のシミュレータを使った発見学習で知識を獲得したと言える。図４はステップS2-1とステップS2-3の学生の知識の差を示した。学生６を除いて、すべての学生で知識獲得することができた。学生６のケースは考察にて議論する。

さらに、我々は前述のステップS3とS4を実施した。ステップS3で教員はレギュラー授業で知識伝達型の講義形式にて学習項目C1-1からC2-2の６項目を発見学習した６名を含む全学生へ講義した。最終テストにて６個の学習項目に関するいくつかの問題を提示した。本科目の全受講生数は70名であり、そのうち６名が数ヶ月前にシミュレータを使った発見学習を行った。６個の学習項目に関して、発見学習を行った６名の成績とそのほかの学生の成績との差を調べると、６学習項目の正解率の差を確認することができた。発見学習に取り組んだ学生の正解率が発見学習に取り組まなかった学生の正解率を上回っていた。両カテゴリの正解率の差はおよそ45%に達した。調査対象問題は10問で、t検定の結果、危険度５%で有意な差を確認することができた。したがって、ソフトウェア工学教育におけるシミュレーションを使った発見学習を予習プ

表１：発見学習による知識獲得

学習	学生1		学生２		学生３		学生４		学生5		学生６	
項目	S2-1	S2-3	S2-1	S2-3	S2-1	S2-3	S2-1	S2-3	S2-1	S2-3	S2-1	S2-3
C1-1	△	△	×	○	×	○	×	×	×	△	×	×
C1-2	×	△	×	△	×	△	×	○	△	△	×	×
C1-3	×	○	×	○	×	○	×	○	△	△	×	×
C1-4	△	△	×	△	×	△	△	△	△	△	△	×
C2-1	△	△	×	△	×	△	×	△	×	△	×	×
C2-2	△	○	×	○	△	○	×	○	×	○	×	×
知識量 (%)	33	66	0	75	10	75	10	66	25	60	10	0

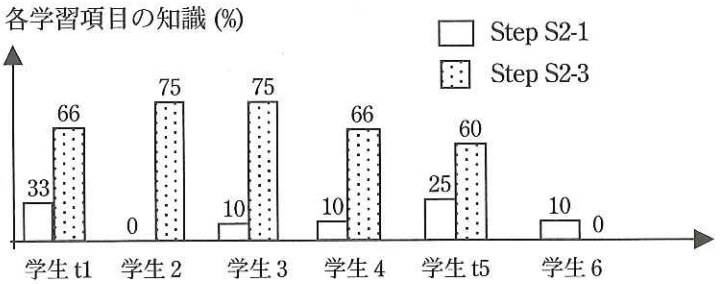

図4：発見学習の結果

ログラムとして取り入れることはソフトウェア工学の知識を効果的に獲得する上で有益であることが確認できた。

5．考察

5．1　HInT上での発見学習の可能性

一般的に発見学習は学生が試行錯誤する時間が必要なので、講義形式の教育よりも時間がかかる。プロジェクトシミュレータは学生の効率的な発見学習実施を支援できる。シミュレータでのソフトウェア工学教育の効果は多くの文献に記されている[1][3][8]。ここで、発見学習におけるHInTの役割について議論する。

すでに述べたように発見学習には非常に長い時間が要求される。同時に、学生は発見学習中に教員へ多く質問して回答を期待する。学生は様々なプロジェクト状況のシミュレーションを繰り返し実行しなくてはならないので、予測しないシミュレーション結果が生じた時に教員への質問を行う。その予期しないシミュレーション結果が不幸にもシミュレータプログラムのバグによって発生した場合は、教員はバグである旨を学生へ早急に伝えなくてはならない。したがって、学生と教員は少なくとも同一場所にいて、即座に質疑応答できる体制を整える必要がある。しかし、長時間の学生シミュレーションに教員がすべて同席できない。学生もほかの授業があり、教員も会議スケジュール等がある。さらに、夜間には学生、教員ともに帰宅しなくてはならい。そこでHInTが有効に活用できる。たとえ教員と学生が遠くに離れていたとしても、インターネットを通じて質疑応答や情報交換、またバグを修正したシミュレータのダウンロードがリアルタイムで可能である。また、HInT上の関係者のコミュニティを作ることで、学生間で有意義な意見交換が可能となり、いっそう効果的な発見学習が期待できる。このようにHInTのインフラが大学に完備していることで、発見学習実施の可能性が高まったと言える。

5．2　HInT上の教育サイクルと発見学習の関係

ここでは、今回の実験で発見学習できなかった学生6について考察する。4.4で述べたよう

に、他学生が発見学習で知識を獲得できたにもかかわらず、学生６は知識を得ることができなかった。学生６は「全く自分がなにをしているのかわからなかった」と感想を述べ、発見学習の学習項目のみならず、プロジェクトの作業割り当てやシミュレーション結果の見方まですべてわからなかった。本稿の“はじめに”で述べたように、この学生６の存在が学力差による教育の難しさの典型的な例である。ある学生はソフトウェア開発に関するある程度の知識があるが、ある学生は全くソフトウェア開発に関する知識がないという状況である。学生６はオブジェクト指向開発のみならず、ソフトウェア開発やプログラミングの知識さえなかった。教員が発見学習という手段を取り、授業をできるだけ工夫したとしても、統一的な教育を実施するだけでは全学生に対しては不十分であることがわかる。

３章ですでに説明したHInT上の教育サイクルはこの典型的な学力差による問題の解決を支援する。この解決の重要なポイントは図２に示す学生カルテの存在である。学生カルテは履修登録、正規科目の成績、エクステンションプログラムの参加状況と成績、個人情報などが含まれる。学生カルテを参照することで、教員は学生の学力を把握することができ、個人の学力に応じた教育プログラムを独自に作成することが可能となる。発見学習で知識を得ることができなかった学生６に対しても、事前に学生カルテを参照することで、個別の予習プログラムを用意することができる。しかし、この個別学習プログラム作成では重要な問題が発生する。ソフトウェア工学授業の受講生は70名である。もし教員が70名のそれぞれの学力に応じた個別学習プログラムを学生カルテを参照しながら手作業で作成したとすると、理論的には可能であったとしても、膨大な時間と労力を要求されるであろう。教員は数科目を担当する場合もあり、現実的には不可能な作業となる。学生カルテを含む論理的な教育サイクルは現実的に実行不可能となり、学力差により発生する本問題を解決することができない。

そこで、HInTは短期間で個人別学習プログラムを作成する環境を物理的に提供する（図５参照）。HInTは論理的な教育サイクルが現実の授業に適応できないという問題を解決する。HInTの中には様々な学力に対応した様々な教材が蓄積されており、学生カルテに基づき、蓄積された教材を使って自動的に個別学習プログラムを生成することが可能である。それぞれの科目に必要とする前提知識をあらかじめ登録しておく。授業が始まる前に、あらかじめ登録された前提知識と受講生の学習履歴と成績（学生カルテより取り出す）の差から、個人別学習プログラムを自動生成する。受講生が足りない前提知識に関する教材をHInTの教材DBより抽出し、それらを組み合わせて予習プログラムを生成する。予習のあとは通常の講義を受講し、その後の復習プログラムに関しても、予習プログラムに対応して不足した前提知識の確認も行う内容を組み込む。反対に十分前提知識のある学生に対しては、予習項目はなく、復習プログラムに関しても講義で受講した内容のみの復習項目となる。多量の教材の蓄積と科目受講のための前提知識を関連づける機能を装備すると自動で個人別学習プログラムの生成が可能となる。

発見学習において知識を獲得することのできなかった学生６に対しても、この個別学習プロ

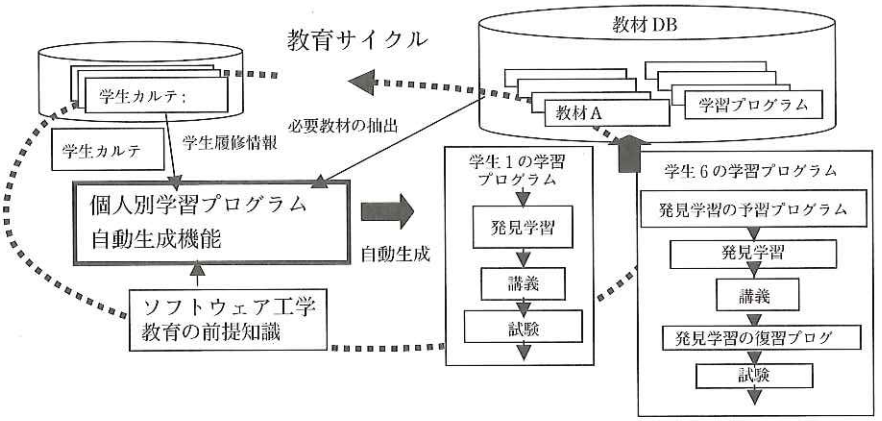

図5：HInT上の自動化された教育サイクル

グラムを適用すれば発見学習の効果はもちろん、ソフトウェア工学教育全体に対しても効果があったと思われる。学生6の場合は、発見学習の前提知識である「ソフトウェア開発」、「プログラミング技術」、「オブジェクト指向開発」などをあらかじめ学習しておくべきであった。「ソフトウェア開発」を学習するために必要とされる前提知識もさらに調査対象となり、もし不足していれば「ソフトウェア開発」を学ぶための予習プログラムもさらに追加されることとなる。このように特定の科目を学習するためには必ず前提知識が要求され、その前提知識が学生の持つ知識と一致するまでリカーシブルに予習プログラムが自動生成される。したがって、前提知識が全くないのであれば、予習プログラムの量は自動的に膨大になっていく。予習プログラムの量が学習不可能なほど大きくなるならば、本受講生はこの科目を受講しないほうがよいという示唆をすることも可能となる。これによって、アンマッチな能力をもつ学生が受講することを避けることができ、本質的な講義内容を確立することができる。さらに、自動生成された個別学習プログラムを再びHInTの教材DBへ登録することによって、個別学習プログラム自身も前提知識不足時の教材検索対象となることができる。このように様々な教材のみならず、様々な学力の学生と科目間の予習復習プログラム自身も登録された教材DBはHInT上の教育サイクルの中核となり、予習復習プログラム生成機能を追加することで、自動学習プログラム生成機能を実現することができるであろう。

6．まとめ

Webベースの統合教育支援システムHInT上に、シミュレータを使った発見学習を組み込んだソフトウェア工学教育の実施例を紹介した。発見学習による知識獲得は、通常の講義形式による教育よりも効果があることが確認できた。さらに、シミュレーションを使うことによって、従来の実体験では数週間単位の発見学習が、7時間から10時間で実施できた例を示した。また、

様々な学力を持つ学生に個別学習プログラムを学生カルテを元に作成する必要があるが、HInTの教材DBを利用することで、個別学習プログラムを自動生成できる可能性があることを考察した。これによって、学生の能力に応じた個人別の予習復習の学習プログラムが教員の多大な労力なしに生成でき、有効であることが考えられる。今後は自動学習プログラム生成機能の実装と共に、学習プログラムの教材DB登録を実現し、ソフトウェア工学科目だけでなく他の科目においても効果的な教育の実現を目指す予定である。

参 考 文 献

[1] Blake, M.B. (2003). A student-enacted simulation approach to software engineering education. *Journal of education, Journal of IEEE Transaction on, Vol.46, Iss.1.* 124- 132.

[2] Bouras, C. Giannaka, E. & Tsiatsos, T. (2003), Virtual collaboration spaces: the EVE community. *Applications and the Internet 2003 Symposium SAINT2003,* (pp. 48-55).Proceedings of the 2003 International Symposium on Applications and the Internet, Orlando, Florida, USA.

[3] Drappa, A. & Ludewig, J. (2000). Simulation in software engineering training. *The 22th International Conference on Software Engineering ICSE2000,* (pp. 199-208). Proceeding of the 22th International Conference on Software Engineering, Linmerick, Ireland.

[4] 花川 典子, 赤澤 佳子, 森 章, 前田 利之, 井上 俊治, 筒井 茂義: シームレス環境を実現したWebベース統合教育支援システムの構築, 電子情報通信学会論文誌, Vol.J88, D-I, No.2, pp498-507, February(2005)

[5] Hanakawa N., Matsumoto K & Torii K. (2002a). A knowledge-based software process simulation model. *International Journal of Annals of Software Engineering, Vol.14,* pp.571-580

[6] Hanakawa N., Matsumoto K., Torii K. (2002b). A communication workload estimation model based on relationships among shared works software development projects. *The 9th Asia-Pacific Software Engineering Conference, APSEC2002,* (pp.571-580). Proceedings of the 9th Asia-Pacific Software Engineering Conference, GoldCoast, Australia.

[7] Hanakawa N., Morisaki S. & Matsumoto K. (1998). A learning curve based simulation model for software development. *The 20th International Conference on Software Engineering, ICSE98,* (pp. 350-359). Proceedings of the 20th International Conference on Software Engineering, Kyoto, Japan.

[8] Oh, E. & van der Hoek. (2002). A Towards game-based simulation as a method of teaching software engineering. *The 32nd ASEE/IEEE Frontiers in Education Conference, FIE2003.* (pp. S2G-13- S2G-18). Proceedings of the 32nd ASEE/IEEE Frontiers in Education Conference, Vol.3, Boston, USA.

[9] Schmid, C. & Ali, A. (2000). A Web-based system for control engineering education. *2002 American Control Conference ACC2000,* (pp. 3463-3467). Proceedings of the 2002 American Control Conference, Chicago, Illinois, USA.

（2005年3月25日受理）