PlayStation によるプログラミング教育について

河野真治
琉球大学工学部情報工学科
科学技術振興事業団さきがけ研究21(機能と構成)領域

Sony のゲーム機である PlayStation を使ったプログラミング教育に関して報告する。ゲーム作成は、リアルタイム・プログラミングの技術だけでなく、グループでのプログラム作成、プログラムの完成度に対する評価などの教育に適している。また、プログラムの技術を形として残し伝えていくことなどのメタな技術を実践することができる。これらの特徴により社会が要請するプログラマの能力と、大学の提供する教育のギャップを埋めることができると期待される。

Programming Education using PlayStation

Shinji Kono
Information Engineering, University of the Ryukyus,
PRESTO, Japan Science and Technology Corporation

Programming Education using PlayStation (Sony's Home Video Game Machine) is presented. By building a complete video game program, not only real-time programming skill but also group working capability and sense of completeness of the product can be achieved. Students are requested o establish meta technology such as technology transfer or making usable library. We hope to fill the gap between social requirements for programmers and university level education of programming.

1　情報工学でのプログラミング教育と生産現場のギャップ

情報関連の進歩の度合いは、他の科目に比べて異常に速く、変化も激しい。情報工学を教える場においても、教師が教えられてきたことをそのまま教えることは、まったく通用しなくなってきている。特に、情報工学卒の人材は、IT 分野あるいはコンピュータ業界、ソフトウェアハウスなどで即戦力として期待されるべきものである。

しかし、最新の知識を持つべき学士あるいは修士であっても、採用された企業での再教育が必要とされるのが現実である。その一つの原因は、大学で教えられるプログラミングが「オモチャのプログラミング」であって、実際の製品レベルのソフトウェアを作成するレベルに到達してないことが上げられる。そのようなレベルのプログラミング技術は、授業で教えられるようなものではなく、人から人へ伝えられるものである。

また、単に技法としてのプログラミングに習熟するだけでは、ソフトウェア生産の現場で要求される、「顧客の要求を理解し、必要なソフトウェアの仕様と、作成プロジェクトを提供すること」には、まったく不十分である。

これは、教える方が経てきた環境に依存するものだとも考えられる。実際、大学教官は半分は研究者であり、そのバックグラウンドは研究である。つまり、実験結果を出すための比較的簡単なプログラムしか体験していないのが現実である。また、プロジェクト・マネジメントなどの経験を持つものは皆無に等しい。

それでは、そのような人材をそろえて、学生に対して実践的なプログラミング技術を教えることが望ましいのだろうか? 実際には、高校を卒業したばかりの学生に、実務レベルのプログラム作成技術とプロジェクト・マネジメントを体験されることが望ましいとは言えない。なぜなら、情報工学に入学した学生は、実は、プログラムというのがどういうことか、まだ、把握しておらず、時間のかかる複雑なプログラミングにとりかかる動機に欠けているからである。

このような学生に対して、いきなり複雑なプログラムに取り組ませることは、むしろ、学生のやる気をそぐことになる。一方で、やさしいプログラム課題ばかりでも、学生のやる気をかきたてることは難しい。やる気とは、実際に達成された目標の繰り返しで出てくるものであり、その目標が低いと、その効果は小さい。

2　新世紀のプログラム教育に要求されるもの

1 年が 7 年に相当する言われる情報分野で通用する人材、研究できる人材を育成することが、企業に取っても大学に取っても必要である。そのためには、以下のような教育が必要であると考えられる。

- ものを作ることを通した動機付け、やる気を作る
- 総合的なプログラミング・プロジェクトの体験させる
- 技術、知識の蓄積方法を身に付ける
- 技術を人から人へと伝える技術を体験する

琉大の情報工学科では、3 年の実験に「ものづくり」を中心とした課題を導入している。本文で報告する「PlayStation 上のマルチメディア・ゲーム作成」もその一部である。この課題の狙いは、

- 具体的な目標を持つ長期的 (半年から 1 年) を持つ課題
- 毎年毎年、新しい技術に触れる機会であること
- 学年を越えて、技術を蓄積し、伝えていくこと
- 学生自らがプロジェクトを立てて実現していくこと

などの機会を提供することである。

実際に、新世紀に要求されているプログラム技術とは、アルゴリズムやデータ構造などの基礎技術はもちろんだが、

- インタラクティブ・プログラム
- リアルタイム・プログラム
- 2次元、3次元グラフィックス、アニメーション
- ネットワーク・プログラミング
- 並列プログラミング

などの進んだ技術、さらに、

- 音楽や音の作成技術
- 絵や写真の処理
- プログラムの実行の管理
- 作成されるプログラムの独創性

なども要求されている。

3　メタ技術としてのプログラム教育

大学教育は、時間的にも限られており、学生はプログラミング技術習得に専念するわけではなく、教養及び語学の授業、他の専門科目が存在する。3年あるいは4年であっても十分なプログラミング技術を身につけるというわけにはいかない。しかし、これは生産現場でも同じであり、完璧なプログラミング技術を持つものだけが仕事をしているわけではない。

プログラミングは、数学の問題を解くのとは異なり、完璧な答が一番良いとは限らない。限られた時間と資源、あるいは人的資源の中で最良の結果を出すことが要求されている。そこでは、新しい技術を習得することそのものの時間も限られている。

従って、

- 新しい開発環境への習熟
- マニュアルやWWWからの情報収集
- 非常に早い開発サイクルへの慣れ

こと自体を技能として習得する必要がある。特に英語の能力が情報収集では重要である。

教師の立場からも、常に最新の情報全てに習熟することは効率が悪い。研究などでは、むしろ深く狭い知識が要求される。しかし、学生に取って重要なのは、広く浅い知識である。学生自身が情報を収集し、大学内部の知識として具体的な形になるように残していくことが学生に取っても教師に取っても利益となる。

さらに、実用規模のプログラミングは、多人数で行われるのが普通であり、学生同士でグループを作る場合でも、能力差、知識差が問題となる。

- ツール作成

- データ作成
- 練習、習熟

などをメンバの状況に応じて割り振る必要がある。また、

- 技術をライブラリや文書の形で保存し、伝えること

も重要である。これらは、個々の人材の固有の技術ではなく、それらを支える、または、技術を作っていくメタな技術であると言える。

また、ゲームなどは、最終的な製品に近く、完成度が問われる。完成度とは、

- ユーザの目に作成されていない部分が見えない
- ゲームの始まりから終わりまでが一通りそろっている
- 絵と音が出る
- あらゆる入力に対して対処がなされている

などで判断される。この判断力そのものを身につける必要がある。また、他のグループのアウトプットに対する評価をする能力も重要である。

これらのメタな技術は、教科書を作成し、教えるよりも、実際の作業を通して見つけていくことが効率的だと思われる。また、予算の限られた大学で、教育が陳腐化するのを防ぐ意味でも、このようなメタな技術を教えることが重要である。

4 モティベーションを作る教育

コンピュータを操作できるだけで、プログラミングができるだけで十分な技術だと思われた時代は既に過ぎ去っている。学生は、既にコンピュータに慣れ親しんでいるのであり、コンピュータにできることに対する漠然としたイメージを持っている。これは、プログラミングするということに対するイメージの欠如と対照的である。

従って、テキスト入出力を中心とした導入は、学生の持つコンピュータのイメージ (ゲーム機や、カーナビなど) と掛け離れている。情報工学科でさえ、テレビゲームを作りたいと言うのを志望理由にあげる学生は無視できない数いるのが現実である。これを幼いと言うことは正しい認識ではない。今の学生に取って、コンピュータとはそのようなものなのである。

自分のイメージから離れた目標に対する動機付けをすることは難しい。もちろん、プログラミングは、派手なロボットの動きやゲームの画面とは掛け離れたものである。しかし、それを乗り越えて、プログラミングそのものに興味を持たせることには、なんらかのきっかけが必要だと思われる。

テレビゲームは、技術と言うよりは芸術である。しかし、技術のない芸術は存在しない。そして、その技術そのものが目的になったのがコンピュータ技術だということもできる。その意味で、テレビゲームからコンピュータを学ぶ動機付けをする方法は、順序としてそれほど間違っていない。

また、ゲームを作ると言うことは、新しい遊びを作ると言うことでもある。それは高度に創作的な作業であり、技術だけはないし、技術からこそ出てくるものでもある。例えば、ボールを投げるという技術がなければ、野球と言う遊びはでてこない。しかし、技術だけならば、それはゲームにはなりえない。

5 PlayStation 開発キット

PlayStation は、95 年に発表された Sony のテレビゲーム機である。

- 2Mbyte Memory
- 37MHz Mips CPU
- Geometry Engine

という、当時 Unix ワークステーション並と言われた構成は、今から見ると、貧弱といって良い。

「ねっとやろうぜ」と呼ばれるカスタマ向けゲーム作成環境は、96 年に発売された。

- パソコン上のクロスコンパイラ
- パソコンと接続可能な特別製 PlayStation
- 3D Lightwave 3 次元モデリングツール
- WWW サーバによるサポート

という構成は、プロ用の機器とは隔たりがある。しかし、基本構成としてはほとんど同じだといって良い。

プロ用の開発機器との差は、ライブラリ内部をいじれないことと、デバッグ環境の差であると考えられる。また、開発キットでは、CDROM を作成することはできず、ダウンロードのみとなる。

PlayStation 自体は、3 万円程度、開発キットは 12 万円であり決して高価とはいえない。組み込みシステムの開発を考えると格段に安いといえる。これらの他に、テレビ、開発用パソコン、インターネットアクセスを用意する必要がある。

コンパイラは gcc であり、Windows だけなく Unix 上でも開発することが可能である。コンパイラを持っていれば、市販の PlayStation でも拡張バスを使うことにより開発をおこなうこともできる。ただし、最新の PlayStation には拡張バスは存在しない。

ソニー SCE は、最初のころに、「ねっとやろうぜ」のサーバを有償化する予定があり、公立大では、そのような年会費のようなものを払う前例がないので購入に問題があった。しかし、結局は、有償化は立ち消えとなり買い取りのみですむことになった。こういう購入時の事務的な問題は、処理できない方にも

6 PlayStation によるプログラミング

PlayStation のプログラムは、テレビの走査線の周期にそってダブルバッファ を用いて、行われる。ダブルバッファとは、二つの画面を切り替えて、表示しているのと別な画面側に書き込むことにより、スムースな表示を可能にする手法。描画は GPU が行い、描画コマンドを登録すれば良い。

```
初期化ルーチン
while(1){
    パッド等、入力データ受け取り
    foreach ゲームオブジェクト{
        ゲーム上のオブジェクトの状態計算
        描画登録処理
    }
    画面切替え操作
    登録描画処理実行開始
}
```

という形をとることが多い。3 次元と 2 次元での本質的な差は、描画登録処理で何を登録するかだけである。

3次元の決まった形は、3D Lightwave でモデルを作り、それを PlayStation に読める形 (TDM 形式) にしてダウンロードする。その形を描画登録処理すれば良い。ただし座標系の計算は3次元なので視点などを含めるとかなり複雑である。線形代数、行列に関する知識は必須である。

このように PlayStation のプログラムは、ハードウェアに密接に結び付いたプログラムであり、限られた時間内に全ての処理を終わらせる必要がある。特に、クロックに依存した形と状態計算からなるので、ハードウェア記述に近い構造を持っている。

プログラムは基本的に C で行われるが、Unix などのスタイルとはかなり異なるものとなる。しかし、同期関連のシステムコールに相当するものは少なく、全体として学ぶべきことはそれほど多くはない。

特に時間の記述に関しては、画面切替えのタイミング (60Hz) に厳密に同期するために、正確な同期を実現することが可能になっている。OS が関与する Direct X や Open GL と異なり、その処理は、決定的な処理であり、より確実なハードリアルタイム処理を実現することが可能になっている。

7 PlayStation によるプログラミング教育

さらに、PlayStation 用に次の4種類のライブラリを作成した。これらは、学生によって作成されたものである。

- 2次元シューティングゲーム専用ライブラリ
- PlayStation 用 Tuple 通信ライブラリ
- VRML 表示ライブラリ
- TMD ダウンローダ

これらを用いて、

- 2次元ゲーム
- 3次元ゲーム
- 3次元ネットワークゲーム

というステップを踏んで1年で、2回プロトタイプを作ると言うのが最初の2年の標準的な課題である。後半2年では、より多くの学生に PlayStation Programming を経験させるために、半年で二つのグループに独立に同様の課題を課した。

8 課題の運営

通常、簡単な例題、あるいは、先輩の作った例題を読むことから始め、同時に、3次元モデリングについても学ぶ。だいたい3か月で最初のプロトタイプが完成する。

1グループ、3-6 人から構成して、課題の提出やレポートはグループ単位で提出ということにした。これは、プログラミング能力差を吸収するためでもある。一人だけ、あるいは、二人とすると、完成度の高いプログラムを作るのには人数が足りず、また、プログラミング能力の低い学生が集まった場合に、課題が破綻してしまう。

全体的な完成度は、1年通して実験したグループの方が当然高い。例えば、オープニングや音などをつけることにかなりの時間がかかってしまう。

しかし、一番重要なのは「どんなゲームにするか」であり、これをまじめに行うためには、数か月かかるようである。従って、独創的なゲーム作成をめざすならば、一年かける方が良い。半年では、かろうじて動くゲームを作成するのが限界である。

また、プログラミングコンテストに、2年続けて参加し、銅賞一つを獲得した。

9 学生のプログラミングの特徴

銅賞を獲得したゲームは、技術的にはそれほどでもないが、オリジナリティのあるゲームであった。これを作成したグループは、扇子なげや中国ごまなどを普段から練習するなど、テレビゲームではない遊びにも凝ったグループであり、プログラミングが技術だけはないことを示している。また、3次元ゲームを考えるのに、紙で作った模型を作成していた。ゲームの内容を考える時には、有効な手法だと思われる。

ゲームを作る技術の低いグループは、ゲームの面白さをシナリオの導入で補うことを考えることが多い。しかし、技術の高いグループが作ったゲームが面白いとは限らない。「すごい」と人を思わせることができるが、それと面白いゲームとは独立なもののようである。

最初に簡単な例題、あるいは、完成したプログラムを渡してしまうので、どうしても、ほとんどのプログラムは、人のをそのまま移したものとなっている。

ライブラリを作ったり、それを人に使わせたりするのは、かなり高度なことであり、ごく一部のグループでしかおこなわれなかった。

特に指導しないと、PlayStation では巨大な main 文一つでサブルーチンがほとんどないプログラムを書くことが多い。これは一つは、PlayStation のプログラム構造が巨大な while 文になっているので、それに引きずられているのだと思われる。

また、例えば、ミサイル5つの処理などを、「一つのミサイルの処理をカットアンドペーストして、5つ並べる」というスタイルが良く見られる。これが、全体の構成をみずらくしてしまう。

また、C のポインタと構造体は、C の初心者の鬼門であり、配列のみを使って、構造体を避けていることが多い。例えば、オブジェクトの位置を、

```
int object_x[5];
int object_y[5];
```

のような二つの配列で持ってしまう。これを、

```
struct object { int x; int y; } object\_list [5];
```

のように書くことができない。従って、malloc() がほとんど出て来ない。

一方で、malloc() のちゃんとした初期化は非常に難しいらしく、malloc() の使用を強制すると、プログラムミスが増え、デバッグが大変になる。だとすれば、配列を使うこと自体は、プログラミング能力に見あった妥協だとも言えるかも知れない。

Playstation は 2MB がすべて高速なメモリというわけではない。従って特に高速なアクセスが必要な場合には特別なメモリ管理をする必要がある。しかし、固定サイズの配列などを局所変数にとることが多くなり、スタックを大量に消費してしまう。これは速度的にもあまりうれしくない。

一部のグループは、Playsttion で動かす前に、Windows 上でプロトタイプを作成している。VC++ + Direct-x を用いることが多い。PlayStation 上でも C++ を用いたグループが一つあった。ただし、C++ を効果的に使っているコードではなかった。

PlayStation 開発キット「ねっとやろうぜ」はドキュメントのバグが多く、それが大きな障害となった。さまざまに、試しながらプログラムしていくようなことが必要になっていた。

Tuple 通信ライブラリを、理解することができず、通信を使った同期を行う方法もわからないので、ネットワークゲームを作ることができない。一つは、ドキュメントが不備だからかも知れない。一つのグループなどは、自分で通信ライブラリを別に自作していた。

10 学生のプログラムの管理

プログラムを版管理することができず、複数のディレクトリに似たようなプログラムがたくさんあり、どれがどう違うのかを説明できない。RCS や CVS を使えば良いのだが使わない。

11 PlayStation 開発キットを使ったプログラミング教育の評価と課題

PlayStation は学生に取って馴染みのあるハードウェアであり、どちらかといえば、ユーザフレンドリでない開発環境であるが、学生の興味をひく教材となっている。「ねっとやろうぜ」キットは値段的にも適当であり、大学生あるいは専門学生に対するイタラクティブ・プログラム、リアルタイム・プログラムの教材としてもすぐれている。

しかし、ハードウェアとしては特殊であり、know how の蓄積が要求されるので、そのための対応が必要である。もちろん、これらの know how の蓄積、技術指導自体も実用的なプログラミングの一部であると考えられる。しかし、学年毎の分離がはっきりしている学校では、「上の学年が、下の学年に技術を伝える」ということが難しいかも知れない。しかし、技術を共有する、広めるという態度が、実社会でも重要なのであり、教師の負荷を軽減する意味でも、学生に指導の経験を積ませるためにも、積極的に「技術を伝える」ことを進めるべきだと思われる。また、それが、その学校自体の無形の財産ともなる。

一方で、技術の進歩は長足であり、PlayStation は、すでに PS2 に取って変わられつつある。PS2 は、PlayStation 互換ではあるが、PS2 としてのプログラムを作成するためには、そちらの技術を、また、新しく学ぶ必要がある。これは、特殊なハードウェアを使う限り避けられない。PS2 の開発キットはカスタマ向きまだ、出ておらず、業務用の開発キットは 100 万円ほどする。

PS2 は、2000 年に発表され、

- 32Mbyte Memory
- 300MHz Mips CPU
- Emotion Engine, VU x 2

という構成をとる。このアーキテクチャは年々陳腐化するのであり、既に、1GHz の CPU がアナウンスされ、256MB のメモリを実装するのがパソコンの標準となっている 2001 年現在、でも能力的に高いとはいえない。しかし、グラフィックスの能力はまだ比較的高い。また、VU と呼ばれる二つの DSP (Digital Signal Processor) の能力はまだ生かし切れているとはいえない。

Direct X あるいは、Open GL を使えば、陳腐化という点では多少問題を避けられる。これら二つは、PlayStation に比べれば寿命が長いと考えられる。また、これらは参考にする書籍が多いことも大きな利点である。一方で、PlayStation は、OS よりも自由度の高いリアルタイムプログラミングを可能としている。また、3 次元表示、画像処理専用のハードウェアであるので、その技術をより直接に学べるという利点がある。PlayStation の方が若干学生にアピールすること、リアルタイム性が高いことを考えて選択すべきであろう。一方で、学習のための閾値が高いことも確かであり、ざっと画像処理に関して勉強したいという学生には不向きであると思われる。

すべての企業が優秀なプログラマを求めているわけではない。強力な技術を持つ人材よりも、おとなしく人あたりの良い営業向きの人材を求めているところも多い。この実験コースは、実践的なプログラマ養成を念頭に置いているが、技術だけを追うような教育が本当に望まれているのかどうかは少し議論の余地があると思われる。

学生の志望も研究者、会社員、公務員と多様であり、実験開始時の学生のプログラミング習熟度もさまざまである。もちろん、ある程度の基礎的な力は要求されるが、これが人の能力差があるのが現実の社会でもあるので、一人一人に同じ程度の基礎知識を要求する必要はないと考えられる。

3年生のための実験は、卒業研究とは異なり、通常の授業の一貫として行われる。課題の内容は、必然的に指導教官の卒業研究のテーマと重なる部分が多い。したがって、さまざまな卒業研究を一つでなく、さまざまに体験できるような仕組みになっている。しかし、これらは卒業研究ではないのであり、あくまでも卒業研究の準備にすぎない。学生に取っては、自分の向いている研究分野を見付ける手段の一つになっていると考えられる。

このような「もの作り」を中心とした実験では、毎週決まった時間内ですべての作業をすませることは難しい。実験時間に関しては、柔軟な対処が必要である。初期の頃は、ツールになれるための時間、理解するための時間などで比較的時間はとられない。しかし、もの作りの最終段階では、要求される完成度に応じた時間が要求されることになる。

また、実際に作成されたものを評価することが重要である。コンテストなどは具体的な目標であり、学生の動機付としてすぐれている。

12 まとめ

PlayStation は学生に取って良く知られたコンピュータである一方で、通常のコンピュータよりも習熟するのは難しいものである。チェレンジングな課題だと言うこともできるだろう。

また、決定的なリアルタイム・プログラミングと状態遷移を中心としたプログラミングであるので、ハードウェアとソフトウェアの中間的なプログラミングともなっており、新しいプログラミング分野であるとも言えるだろう。

このようなシステムを使った実験課題を導入することにより、学生の興味を維持し、学習意欲を高め、新しい概念を学習するという力を付けさせることができると考えている。