データベースシステム 113－16
（1997. 7. 14）

知識ベースの可視化機構を利用したアニメーションエディタの設計と実装

柳沢 豊　塚本 昌彦　西尾 章治郎

大阪大学大学院工学研究科情報システム工学専攻
〒565 大阪府吹田市山田丘 2-1
{kani,tuka,nishio}@ise.eng.osaka-u.ac.jp

あらまし：コンピュータ上でのアニメーションを作成するシステムは，計算機を用いた教育やプレゼンテーションなどの目的で広く利用されつつある．このような中で筆者らは，アニメーションのシナリオを知識ベースを用いて管理し，自然言語風のシナリオ記述から知識ベースの推論機構を用いてスケジュールを作成することのできるアニメーション開発システムの開発を行なってきた．本研究ではこのシステムをさらに拡張し，知識ベース内でシナリオから作成されたスケジュールを，グラフィカルユーザインタフェースシステムを用いて可視化することで，アニメーションを自由に編集することのできるエディタの設計と実装を行なった．これを用いることにより，開発者はアニメーションのシナリオを視覚的に把握し，直観的な操作でアニメーションの編集を行うことが可能となった．

キーワード：オブジェクト指向プログラミング言語，知識ベース，マルチメディア，アニメーション

Design and Implementation of an Animation Editor using Mechanisms for Knowledge-base Visualization

Yutaka YANAGISAWA　Masahiko TSUKAMOTO　Shojiro NISHIO

Department of Information Systems Engineering
Graduate School of Engineering, Osaka University
2-1 Yamadaoka, Suita, Osaka 565, JAPAN
{kani,tuka,nishio}@ise.eng.osaka-u.ac.jp

Abstract: In recent years, computer animation systems are used for many purposes such as computer aided instruction (CAI) and presentation. For these purposes, in our previous paper, we have proposed an animation development system where scenarios are described separately from programs and managed in a knowledge-base system. In this paper, extending this system and using the techniques that we will propose for knowledge-base visualization, we will design and implement an animation editor for modifying animations through graphical user interface systems. The developed system enables users to handle easily schedule data for an animation. Moreover, users can modify an animation flexibly and directly using graphical user interface systems.

key words: Object-Oriented Programming Languages, Knowledge-base, Multimedia, Animation

1 はじめに

近年，人間にとって直感的に理解しやすい画像や音声など人間の視覚や聴覚に直接働きかけるデータを統合的に扱うことのできるマルチメディアシステムが，電子辞典や教材，WWW上での情報発信，また計算機を使ったコミュニケーションやプレゼンテーションを支援するシステムとして広く用いられている．これらの応用システムにおいて，キャラクタの動作によって視覚的および聴覚的に人間に情報を伝えることのできるアニメーション技術は，極めて効果的な情報表現手法のひとつであり，現在アニメーションの開発手法に関する研究が数多く行われている [1][2][3][4][5].

このような中で筆者らは，アニメーション作成者が記述した自然言語風のシナリオ記述を知識ベースにおける推論機構を用いて解析し，知識ベース中でルールを用いて詳細なアニメーションのスケジュールに変換するという手法を用いるアニメーションシステム Drake の開発を行ってきた [10][11]. このシステムにおいては，スケジュールデータは知識ベースシステム内に蓄積され，これとは独立に作成されたアニメーションの表示システムが必要に応じて問合せを行ない，アニメーションの表示に必要となる詳細な情報を表示システムへ返すという方式を用いている．この知識ベースへの問合せ操作を自動的に行い，またその問合せの結果からアニメーションの表示に必要な情報を抽出して実際のアニメーションの動作に効果的に反映するための手法として，筆者らが提案している演繹オブジェクト指向プログラミングとよぶプログラミング手法を用いている [8][9].

これまでに提案したシステムを用いてアニメーションを作成する場合には，アニメーション中に登場するキャラクタやその動作を常にシナリオ記述という形で与え，これを知識ベースに付属したパーザを用いてスケジュールデータに変換させるという操作を行う必要があった．そのため，作成されたアニメーションを修正する場合には，基本的にはシナリオ記述を編集するという形でしかアニメーションの編集を行えなかった．しかし，アニメーションの編集をより効率的に行えるようにするためには，シナリオ記述の編集によってのみアニメーションが行えるだけでは不十分であり，知識ベース内に蓄えられたスケジュールデータを直接的に編集したり，グラフィカルユーザインタフェースなどを用いて視覚的にアニメーションを編集することを可能とするような機能が必要であると考えられる．

そこで本研究では，作成者の意図をより詳細に反映させることが可能なアニメーションの開発を支援するシステムを実現するために，知識ベース内に蓄えられているスケジュールデータを可視化し，これを視覚的に編集してアニメーションの修正や変更を行うための機構を実現した．またこの機構を用いて，アニメーションが実際に表示されている間に，ユーザが表示中のアニメーションに対して直接的に行った操作を，知識ベース内のスケジュールデータに反映させることで，アニメーションの編集を直感的に行うことを実現するシステムの設計および実装を行った．

以下，まず 2 章において，アニメーションシステム Drake の概要について述べる．次に 3 章では，知識ベース内に蓄えられたスケジュールデータの可視化手法と，それを用いてアニメーションの編集を行うための機構について述べ，4 章においてこの機構を用いて作成したアニメーションエディタの実装と動作例について述べる．最後に，5 章において本研究のまとめを行なう．

2 アニメーションシステム Drake

本章では，筆者らがこれまでに開発を行ってきたアニメーションシステム Drake (Dynamically Restructurable Animation system using Knowledge-basE) [10][11] の概要について述べる．

Drake は，知識ベースを核にもつアニメーションのシナリオの管理のためのシステムと，実際にアニメーションの表示を行うシステムから成る．シナリオ管理システムとアニメーション表示システムとは，互いに独立したシステムとして作成されており，現在は表示システムとしては Java を用いて実装されたシステムと，VRML2.0 ブラウザ上において表示を行うシステムの実装の二つがある．シナリオ管理システムとしては，現在演繹システム DOT[6][7] をベースとした実装がある．また Drake は，双方のシステムの独立を高め，その再利用性を向上させるための手法として，は筆者らが提案している演繹オブジェクト指向プログラミングと呼ぶプログラミング手法を用いて実現されている．

演繹オブジェクト指向プログラミング (DOOP: Deductive Object-Oriented Programming) [8][9] とは，オブジェクト指向プログラミング言語を用いて作成

したソフトウェアの構造に関する情報を知識として知識ベースに蓄えておき，それらの知識を動的に変更することによってソフトウェア中の構造を動的に変化させることができる，プログラミング手法である．演繹オブジェクト指向プログラミングでは，オブジェクト指向プログラミングにおけるクラスの階層情報や，クラスの構造情報，オブジェクトの参照や関連に関する情報，変数の値やメソッドの動作条件に関する制約などの情報などを，知識ベース中に知識として記述しておく．プログラミング言語を用いて記述する実行コードとしては，プログラム中のクラスのメソッドにあたる部分のみを記述する．ソフトウェアの構造に関する知識を知識ベースで管理することによって構造記述の抽象化を図ることや，構造情報間の矛盾の検出，構造の変化に伴う相互作用の定義などを演繹体系を用いて行うことができるため，ソフトウェアの保守作業の効率を高めることができる．

さらに，演繹オブジェクト指向プログラミングにおいては，ソフトウェアを開発するために用いるプログラミング言語の種類や，利用しようとする知識ベースの種類に関わらず，どのプログラミング言語で作成したソフトウェアからでも，どの知識ベースでも利用することが可能であるという利点をもつ．プログラミング言語の実行時システムと知識ベースとの間の通信方式を統一化するための機構をそれぞれのプログラミング言語，知識ベースごとに用意してあり，用いるシステムの種類の相違によって生じるインタフェースの違いを吸収することができる．

このような特長をもつ演繹オブジェクト指向プログラミングを用いてアニメーションシステムを作成することにより，幾つかの利点が得られる．まず，動的な知識の変更への対応が可能なため，メディアデータの表示スケジュールを知識と見なし，スケジュールからのアニメーション表示に演繹オブジェクト指向プログラミングの機構を利用することによって，アニメーション表示中における動的なスケジュールの変更をアニメーションの内容に反映することが可能である．これによって，ユーザはアニメーション実行中にインタラクティブにその内容を変更することが可能となる．また，演繹オブジェクト指向プログラミングを用いると，知識ベースであるシナリオ管理系とそれを利用するソフトウェア，すなわちアニメーション表示システムとの間の高い独立性を保てることから，ユーザは用いる知識ベースの種類や表示系の種類を，最もユーザが利用しやすいものを選んで用いることが可能である．

3 アニメーションエディタ

本章では，アニメーションシステム Drake を拡張したアニメーションエディタについて述べる．まず 3.1 節において可視化を行う知識についての議論を行い，次に 3.2 節でシナリオ記述から知識ベース内の演繹システムを用いて生成されたスケジュールデータを，グラフィカルユーザインタフェースを用いて可視化し，視覚的に編集することのできる機構について述べる．

3.1 知識の可視化

一般に計算機上でアニメーションを作成する際には，アニメーションに登場するキャラクタとその動作に関してさまざまな情報が必要となるが，これらの情報は次に示すように大きく三つの種類の情報として分類できる．

動作主体: アニメーションに登場するキャラクタ

動作の種類: キャラクタが実際に行う動作

動作の付帯的情報: 動作を行うときの画面上での位置，速度，向きなどといった，動作の補助的な情報

これらの情報は時間軸に沿って刻々と変化し，その情報の変遷に従って表示するキャラクタや，キャラクタの動作を視覚的に表示していくことで，実際のアニメーションが作成される．

Drake においては，アニメーションの開発者は自然言語風のシナリオ記述言語を用いてアニメーションのシナリオを記述する．このシナリオは，Drake のシナリオ管理システムに付属するシナリオパーザによって「動作主体」「動作の種類」「動作の付帯的情報」のそれぞれ解され，シナリオ管理システム内の知識ベースにスケジュールデータとして貯えられる．

このようにしてスケジュールデータが蓄えられた知識ベースに対して，アニメーションの表示系は，ある時刻における「動作主体」「動作の種類」「動作の付帯的情報」に関する問合せを行い，実際のアニメーションを表示する．つまり，これら三種類の情報を可視化し，ユーザが直接的に編集することのでき

```
男の子 は 子ども である
女の子 は 子ども である
太郎 は 男の子 である
花子 は 女の子 である
はじめ 太郎 は 右へ 歩く
はじめ 花子 は 左へ 歩く
挨拶した 太郎は 上へ 歩く
挨拶した 花子は 下へ 歩く
子ども が 子ども に 会った ならば 子ども は 子ども に 挨拶する
挨拶の 言葉は こんにちは である
```

図 1: シナリオ記述

```
Boy < Child
Girl < Child
Taro < Boy
Hanako < Girl
Time0.Taro.action < walk
Time0.Taro.action.direction < right
Time0.Hanako.action < walk
Time0.Hanako.action.direction < left
Time0.Hanako.location.x < min
Time0.Hanako.location.y < half
Time0.Taro.location.x < max
Time0.Taro.location.y < half
...
```

図 2: スケジュールデータの一部

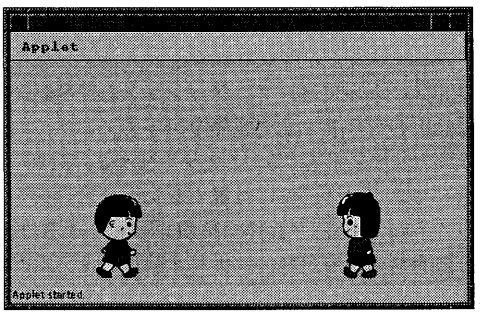

図 3: アニメーション表示例

る機構を提供することで，アニメーションの編集が効率的に行えるようになると考えられる．また，これらの情報の時間的変化を把握しやすい可視化機構を提供することで，アニメーションのシナリオの流れがより把握しやすくなると考えられる．

3.2 スケジュールデータ可視化機構

3.1 節での議論に基づき，本研究では以下の三つのスケジュールデータの可視化のための機構を開発した．

1. スケジュールの流れを時間軸に沿って表示する機構 (図 4).
2. 選択した動作主体に関連するデータを表示する機構 (図 5).
3. ある時刻におけるスケジュールデータの表示を行う機構 (図 6).

図 1 は，シナリオ記述の例である．そのシナリオ記述から，知識ベースにおいて作成されたスケジュールデータの一部を図 2 に示す．また，これらのデータを用いて実際に表示されているアニメーションの実行画面を図 3 に示す．以下，このアニメーションを例にとり，それぞれの機構の動作について説明する．

まず図 4に示した機構は，時間軸に沿った各動作対象の動作の変化を表示させるものである．このウィンドウでは，画面の最上部がアニメーションの開始時刻を示し，最下部がアニメーションの終了時刻を示している．中央左側の列には “Taro” の動作の変化が，また中央右側の列には “Hanako” の動作の変化の様子が示されている．列中にある矩形で囲まれた文字列は，動作主体の行う動作の種類を示し，矩形の示されている位置からその動作が開始され，矢印の終端までその動作が連続することを意味する．動作名が表示されている矩形上でマウスのボタンを押すことで，その動作の付帯的な情報を示すウィンドウが新しく表示される．それが図 5に示す機構である．この機構は，アニメーション上で動作しているキャラクタの絵を直接マウスでクリックした場合にも呼び出される．このウィンドウは，動作主体や動作に対する付帯的な情報を表示するものであり，この例では “Taro” の初期動作である walk に関する情報や，“Taro” 自身の表示色 (服の色) など属性が表示されている．

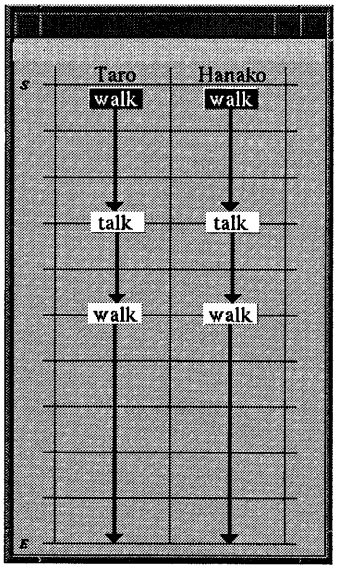

図 4: スケジュール可視化機構 1

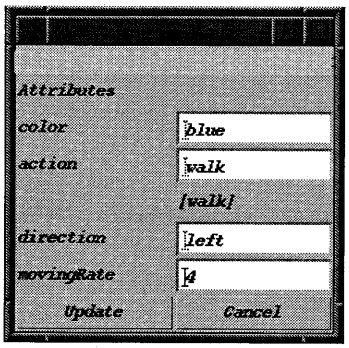

図 5: スケジュール可視化機構 2

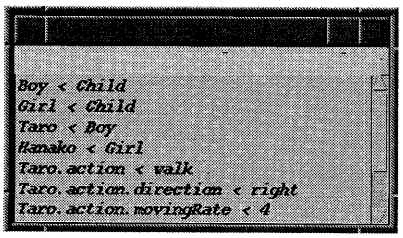

図 6: スケジュール可視化機構 3

また図 6に示す機構は，ある時点における知識ベースから得られるスケジュール情報を直接的にすべて表示するものである．

これらの情報は，ウィンドウ上で直接変更することが可能である．この情報の変更を行う上で，変更の影響を与える時間的範囲を指定することができる．たとえば図 4 において Taro の最初の動作である walk の付帯的情報のひとつである direction の値を left から right に変更した場合，その影響の波及範囲を 1) その動作終了時まで，2) 指定する長さの時間範囲で，3) ある条件が満たされるまで，4) その時刻においてのみ，の四つの範囲指定方法によって指定することができる．この操作は，情報更新時に表示されるパネル上で行うことができる．

なおアニメーションのスケジュールは，アニメーションを実際に動作させる過程でリアルタイムに作成されるため，アニメーションの実行時間全体の中での動作の変遷を表示するためにはアニメーションを一度すべて表示させる必要がある．

4 システムの実装

3 章で述べた機構を用いて開発したアニメーションエディタの概要を図 7 に示す．アニメーションエディタは，従来の Drake にのシナリオ管理システムに 3 章で述べた知識ベース内のスケジュール可視化のための機構を加え，またアニメーション表示部においてユーザからの入力デバイスを通して行われた操作をイベントとして受け取り，それに応じて知識ベースに対してスケジュールの変更を行うための，イベント取得部を追加することで実現した．

シナリオ管理システムに用いる知識ベースシステムとしては，Drake の実装に用いられている DOT [6][7] をベースとして用い，アニメーション表示部，知識ベース内部のスケジュールデータの可視化部およびエディタのイベント取得部は Java を用いて実装した．

5 おわりに

本研究では，知識ベースの可視化手法を利用してアニメーションのスケジュールを視覚的に編集することのできる，アニメーションエディタの設計および実装を行った．このアニメーションエディタを用いることで，ユーザはシナリオ記述から作成されたア

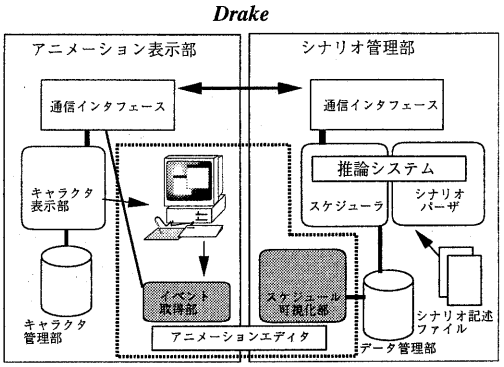

図 7: アニメーションエディタ

ニメーションのスケジュールデータの把握が容易となり，またアニメーション編集をグラフィカルユーザインタフェースを用いて直感的に行うことが可能となった．これにより，従来のシステムに比べ，ユーザの意図するアニメーションを作成することが，より効率的に行えるようになったと考えられる．今後の課題として，開発したシステムを用いて実際にさまざまなアニメーションを作成し，開発効率などの評価を行うことが挙げられる．

謝辞

末筆ながら，本研究の実装に際して貴重な助言および助力を頂いた，西尾研究室の諸氏に感謝する．また，本研究の一部は，日本学術振興会未来開拓学術研究推進事業における研究プロジェクト「マルチメディア・コンテンツの高次処理の研究」によっている．ここに記して謝意を示す．

参考文献

[1] 安倍 広多, 松浦 敏雄, 谷口 健一, "C++を拡張したアニメーション記述言語とその処理系," 情報処理学会大会第 45 回全国大会論文集, pp.353–354(1992).

[2] 舟渡 信彦, 吉川 耕平, 花田 恵太郎, 宮本 雅之, "日本語シナリオからのアニメーションの作成," 情報処理学会論文誌, Vol.34, No.6, pp.1258–1267(1993).

[3] 浜田 浩行, 阪本 清美, Zurowski, J., "インタラクティブアニメーション記述言語とその処理系「AV-Script」," 情報処理学会大会第 45 回全国大会論文集, pp.355–356(1992).

[4] Koved, L., Wooten, W. L., "GROOP: An Object-Oriented Toolkit for Animated 3D Graphics," ACM SIGPLAN NOTICES, Vol.28, No.10, pp.309–325(1993).

[5] 高橋 伸, 宮下 健, 松岡 聡, 米澤 明憲, "アルゴリズムアニメーション作成システムにおける宣言的記述方法について," コンピュータソフトウェア, Vol.11, No.6, pp.83–94(1994).

[6] 塚本 昌彦, 西尾 章治郎, "ドット記法と IS-A 関係を用いた知識表現システム DOT," 人工知能学会誌, Vol.10, No.2, pp.124–133(1995).

[7] Tsukamoto, M., Nishio, S., "Inheritance Reasoning by Regular Sets in Knowledge-base with Dot Notation", Deductive and Object-Oriented Databases (Ling, T. W., Mendelzon, A. O., and Vieille, L., Eds.), Lecture Notes in Computer Science 1013, Springer-Verlag, pp.247-264 (1995).

[8] 柳沢 豊, 塚本 昌彦, 劉 渤江, 西尾 章治郎, "知識ベース独立のための演繹オブジェクト指向プログラミング," 人工知能学会誌, Vol.10, No.6(1995).

[9] Yanagisawa, Y., Tsukamoto, M., Nishio, S., "Deductive Object-Oriented Programming for Knowledge-base Independence", Deductive and Object-Oriented Databases(Ling, T. W., Mendelzon, A. O., and Vieille, L., Eds.), Lecture Notes in Computer Science, Springer-Verlag, pp.345–362(1995).

[10] 柳沢 豊, 坂根 裕, 塚本 昌彦, 西尾 章治郎, "柔軟性の高いシナリオ記述が可能なアニメーション開発システムの設計と実装," 1997 年電子情報通信学会総合大会論文集 (3), p.178(1997).

[11] 柳沢 豊, 坂根 裕, 塚本 昌彦, 西尾 章治郎, "演繹オブジェクト指向プログラミングを用いたアニメーション開発システムの設計と実装," 電子情報通信学会技術研究報告, DE97-4, Vol.97, No.36, pp.19–24(1997).