データベースシステム 121－22
情 報 学 基 礎 58－22
（2000．5．26）

対話的マルチメディア情報提示システム実現のための QUIK の拡張

杉本健二† 緒方啓孝* 的野晃整† 平野尚孝† 横田一正‡ 國島丈生‡
†岡山県立大学大学院情報系工学研究科
‡岡山県立大学情報工学部
*(株) 管理工学研究所

近年, ネットワーク資源は急速な勢いで増加しつつあり, それに含まれる情報は電子化された文書の他に静止画, 動画および音声など様々なメディアや種々の言語によって作成されたアプリケーションなど多種多様化されてきている. このような多様な情報源を統合し, 有効に利用するためのシステムが重要となってきている. そこで我々が研究開発している, 演繹オブジェクト指向パラダイムに基づいた知識表現言語 QUIK を用い, それらの多様な情報源を統合し有効利用しようと考えている. そこで, 本論文ではこれまでの英数字などの記号で記述されたオブジェクトしか対応していなかった QUIK を, マルチメディア情報を扱うための拡張機能を導入すると共に, その応用例として本学に所蔵されているデァドラ伝説の戯曲を視聴覚化するシステムについて報告する.

Extensions of QUIK for Interactive Multimedia Presentation Systems

Kenji Sugimoto† Nobutaka Ogata* Akiyoshi Matono†
Natotaka Hirano† Kazumasa Yokota‡ Takeo Kunishima‡
† Okayama Prefectural University, Graduate School of Systems Engineering
‡ Okayama Prefectural University, Faculty of Computer Science and System Engineering
* Kanrikogaku Kenkyusho, Ltd.

With rapid advances of computer and network technologies, many documents have been electronized, stored, and dispatched over network. Most electronic documents include not only text but also multimedia information such as images, movie, 3D objects, and sound. To integrate such information, it is indispensable to represent and synchronize multimedia information. QUIK is an extension of an deductive object-oriented database language, *Quixote*, for mediating various information over network. Further, we have extended QUIK to treat multimedia information and implemented an interactive drama presentation system, where a Celtic legend, *Deirdre*, is taken as an example. Even if a drama is written in the form of plain text, it has multimedia information essentially. In this paper, we report its prototype system and discuss interactive multimedia presentatiion system.

1 はじめに

1.1 研究の背景

インターネットの普及に伴いネットワーク資源は急速な勢いで増加しつつあり, それに含まれる情報は電子化された文書の他に静止画, 動画および音声など様々なメディアや種々の言語によって作成されたアプリケーションなど多種多様化されてきている. このような多様な情報源を統合し, 有効に利用するためのシステムが重要となってきている.

そこで我々は, 異種分散情報源の統合を行うシステムとして, 演繹オブジェクト指向パラダイムに基づいた知識表現言語 QUIK の研究開発を行っている. これまでの QUIK では, 情報源として英数字などの記号で記述したオブジェクトを対象としていた. しかし, 多様な情報を考えると静止画や音楽と言ったマルチメディア情報を扱うことは自然なことである. そこで QUIK でマルチメディア情報を扱うための拡張機能を設計し実装したので報告する.

2 マルチメディア情報の表現

今回 QUIK で対象としているメディアは, 7 種類ある. これらのメディアは, メディア特有の詳細データと共通する共有データの 2 種類がある. 各メディアは次のようなデータを持っている.

- **type:** メディアの種類.
- **再生時間:** メディアを再生する時間.
- **同期時間:** 他のメディアとの同期をとるための時間.
- **詳細データ:** 詳細なデータを格納.

2.1 対象メディア

まず, 7 種類のメディアとそれぞれの詳細データは表 1 のようになっている.

静止画は gif, jpg および bmp ファイルなどの静止画, 動画は mpg, avi および mov ファイルなどの動画である. オブジェクトは class, wrl の 3 次元オブジェクトを扱うものである. テキストと発話は, 共にテキストを入力するが, テキストはテロップを表示し, 発話は発話処理される. サウンドファイルは wav, mid などの音声およびサウンドを扱う. 移動情報は静止画, 動画および 3 次元オブジェクトの移動情報を指定する.

表 1 において, URL は各メディアが格納されている場所, 文字列は表示または発話させるべき文字列, 表示位置は各メディアの初期表示位置, 縮尺は各メディアの表示の際の縮尺を表している.

2.2 時間の種類

ここでは, メディアに対する時間の種類について考える. 我々は QUIK で取り扱うメディアに対して, メディア時間, 再生時間, 同期時間の 3 種類の時間概念を考えている.

i) **メディア時間**
メディアそのものが持っている時間. 例えば動画などの場合, 始まって終るまでの時間を意味している. 静止画の場合は, メディア時間ゼロまたは無限大と考える.

ii) **再生時間**
実際にメディアを表示する時間. この時間によって, 静止画を表示させる時間を設定できる.

iii) **同期時間**
処理するメディアが複数ある場合に, メディア間の同期をとるための時間である.

それぞれの時間についてもう少し具体的に説明する. 例えば, 表 2 で示すような 3 つのケースで, 静止画 a と動画 b をある画面に表示させることを考える.

まず, ケース 1 の場合では, 図 1(1) のように静止画 a の再生時間が無限大で動画 b の再生時間が n なので, 静止画 a を表示してから静止画 a が消去されずに動画 b が n 時間だけ表示される. なお t1 と t2 は, 画像を表示するための時間である. もし, 画像を表示させる処理系

表 1: 各メディアの詳細データ

type	種類	URL	文字列	表示位置	縮尺	その他
1	静止画	○		○	○	
2	動画	○		○	○	
3	オブジェクト	○		○		
4	テキスト		○			フォント, 大きさ, 色
5	移動情報					対象メディア, 移動情報
6	サウンド	○				音量
7	発話		○			フォント, 高さ, 早さ

がネットワーク上の別の場所に存在した場合では, 通信時間も t1 に含まれる.

次にケース 2 の場合では, 図 1(2) のように静止画 a の再生時間が m で動画 b の再生時間が n なので, 静止画 a を m 時間だけ表示し静止画 a を消去してから動画 b を n 時間表示する.

最後にケース 3 の場合では, 図 1(3) のように静止画 a の再生時間が m で動画 b の再生時間が n で, 更に静止画 a に同期時間 l があるので, 静止画 a を表示した後, 動画 b を表示する. つまり, 時間 t1 + l から静止画 a と動画 b は $m - l -$ t2 時間同時に表示されることになる.

表 2: メディアの時間

ケース	メディア	メディア時間	再生時間	同期時間
1	静止画 a	無限大	無限大	0
	動画 b	n	n	0
2	静止画 a	無限大	m	0
	動画 b	n	n	0
3	静止画 a	無限大	m	l
	動画 b	n	n	0

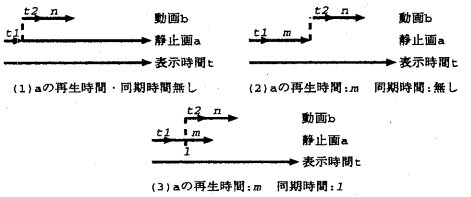

図 1: 各時間によるメディアの表示

メディア間の制御については, ストリームデータを同期させる SMIL(Boston)[8] や TVML[9] のようなものが考えられるが, 本システムでは対話性を実現するために上記仕様にしぼった.

2.3 シーングラフによるマルチメディア情報の制御

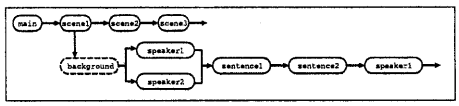

図 2: シーングラフの概念図

シーングラフとは, シナリオの実行順序をグラフィックで表示したものである. 図2にシーングラフの概念図を示す. マル (後ノードと呼ぶ) の中は, 各メディアの名前を示しており, 破線で表現されたノードは, 現在実行しているノードを表している.

ユーザは, クライアントの再生, 一時停止, 停止ボタンを使用しビデオのような感覚でシナリオの制御が行なえ戯曲を楽しむことができる. 再生はシーングラフの実行順序をたどりノードに対応したメディアを次々に再生していく. 停止は, 再生中のシナリオを停止させる. 停止後に再生させると, シナリオの最初からシナリオが開始される. 一時停止は, 再生中のシナリオを一時停止させることが可能となる. そして, 再度再生を行なえば一時停止を行なったシーングラフのノードからシナリオが再開される. また, 移動情報等の付加情報の編集は, 一時停止してからデバックすることになる. さらにシー

ングラフの任意のノードをクリックすることで, そのノードからシナリオを再生することが可能となる.

しかし, 任意のノードから再生すると次のような問題点が考えられる. 例えば, 図3のシナリオを考える.

```
1:  ラヴァーカムが中央に座っている。
2:  老女が左手から登場する。
3:  老女が話す。
```

図 3: シナリオの例

図3のシナリオをシーングラフにすると図4のようになる. (行番号をノード名とする.) なお, 現在処理されているノードは 1 である.

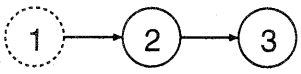

図 4: シーングラフ

ここでユーザが 3 のノードをクリックした場合, 2 のノードを飛ばしてしまうので, 老女のオブジェクトが現れること無く老女の声が再生されてしまうと言った情報の欠落の問題が生じてしまう. この問題を解消する方法として次の方法を考えた. 先ほどの例で考えれば, ユーザが 3 のノードを選択しても, ホストは 2 の情報もクライアントに送信を行なう. その情報を受けとったクライアント側は, その情報が音声の場合は再生せず, 画像であれば再生時間をゼロにして表示することで早送りすることを考えている. また, 現在実行しているノードより前のノードを選択した場合もホストは, 始めからの情報をクライアントに送信し, クライアントは早送りと同様の処理を行なう. 指定されたノードまで実行されたら通常の処理を行なう.

この方法では, 飛ばしたノードの個数が多かったり, シナリオの終りの方で一つ前のノードを選択した場合など, 無駄な処理が多くなるためアルゴリズムの改善が今度の課題である.

3 システムの実現

3.1 全体像

マルチメディア機能の導入の応用例として, QUIK をケルト文学のディアドラ伝説を視覚, 聴覚化する戯曲提示システムの中核をなす実行エンジンとして利用した.

戯曲を QUIK プログラムに変換するための戯曲編集システムを導入し, 従来からある QUIK システムに 図 5 で示すような戯曲を提示するためのマルチメディア情報を扱う機能を大幅に拡張している.

戯曲を編集するためのシステムは, 2 つの処理系で構成されている.

構造化システム
テキスト形式で記述されている戯曲を XML 形式の記述に変換する.

ト書きエディタ
構造化された XML ファイルのト書き部分の編集を行なうためのエディタで, 画像や音声ファイルの指定など具体的内容に変更する.

サーバの処理としては大きく分けて 2 つの部分で構成される.

言語変換系
構造化された戯曲を QUIK プログラムに変換する.

QUIK エンジン
クライアントの要求に従って, QUIK プログラムから処理を行なう.

クライアントの処理系は次の 5 つの部分で構成される.

ユーザインタフェース
ユーザからの要求や入力を受け付け, サーバからの返信を表示する.

サウンド処理系
サウンドファイルの再生や, 発話による発音や消音等の処理を行なう.

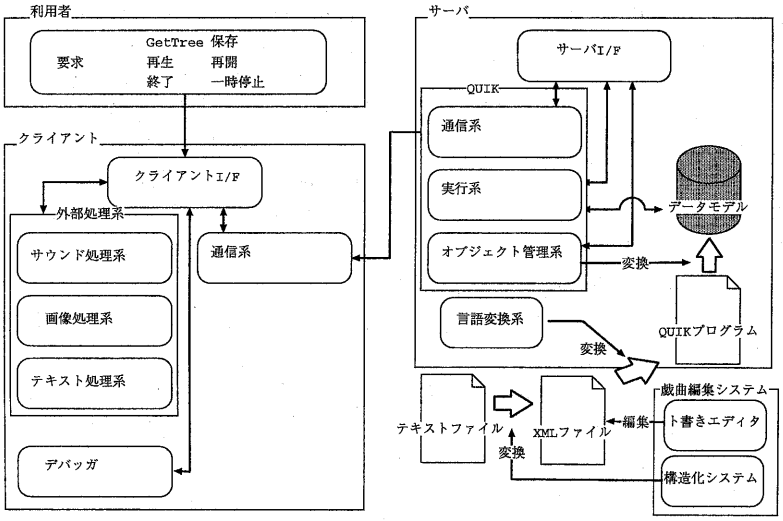

図 5: 戯曲提示システム

画像処理系
静止画, 動画やオブジェクト等の描画や画像の消去, それらの移動等を行なう.

テキスト処理系
テキストの表示や消去を行なう.

デバッガ
ト書きから得られなかった移動情報, 描画サイズや再生時間などの情報の修正を行なう.

従来の QUIK システムの通信は, Java 言語で記述された移動エージェント Aglets を使用している. 今回, PC 側 (Windos98) のクライアントは Java3D 等の機能を使用するため, Java2(JDK1.2) で開発した. しかし, Aglets は Java2 に対応していなかったため通信系に改良が必要となった. そこで, ユーザインタフェースと通信マネジャーの間にダミークライアントを追加し, この問題を解決することにした.

3.2 外部制約解消系としてのメディア処理

論理型言語の観点から見れば, QUIK には項に対応するオブジェクト間の包摂制約に基づいた制約論理型言語である. しかし, 様々な応用を考えれば包摂制約以外の制約の導入が要求される.

QUIK では, 外部制約解消系を QUIK プログラムとは独立させ, 独立したメディエータとして扱うことにする. これによって,

- 複数のメディエータ (QUIK プログラム) が 1 つの制約解消系を共有する.
- オントロジーによって制約解消系を選択できる.

などの長所がある.

また各メディアにメディエータ識別子を導入している. メディエータ識別子を導入することによって, 他のメディエータにメディアを探索することが可能になる. 従って, 一つのシナリオを記述する際にシナリオを構成する全てのメディアが一つのサーバに存在する必要性が無くなる. メディエータ識別子を導入することによって, さらにメディアの共有が可能となる.

ここで, 他のメディエータに問合せと出す方法は, 従来の QUIK の問合せを用いる. しかし, 様々なメディエータに問合せを出すことによって求められたメディアが一意に決まるとは限らない. そこで, 構文の属性に変数を導入し, 制

約によって求めるメディアを一意に決めることにしている.

3.3 マルチメディアの内部データ構造

QUIK において, マルチメディア情報に関するオブジェクトは, リスト構造の形で格納されている. マルチメディア情報の提示に対話性のある演出機能を持たせるためには, この内部データ構造への変更が必要となる. そのためにこの内部データの更新は論理的にはスタック構造で実現され, 必要に応じてロールバックできるようにしている. この変更にユーザの指定による永続性ももたせることができる.

サーバからクライアントにメディアオブジェクトを送信しようとした場合, JAVA の特性として送信しようとするオブジェクトにつながる全てのオブジェクトを送信しようとしてしまう. これでは, 不必要なデータをクライアントに送信してしまい, ネットワークのトラフィックが低下し, 応答速度が極点に低下してしまう. そこで, 目的のオブジェクトと同じものを生成し, そのオブジェクトから不必要なリスト構造を切り離し, クライアントへ送る方法をとっている. また, クライアントは独自の JAVA クラスファイルと, サーバにある最低限の JAVA クラスファイルしか持っていない. これは, サーバ側の QUIK の処理系等, クライアントとして不要なファイルを極力減らすことでクライアントとして必要なプログラムファイルの容量を減らそうと考えるものである. 通信を行なう場合, 送信されるオブジェクトの継承関係の親に当たるクラスファイルはクライアントでも持っていなければならない. しかし, この継承関係の親に当たるクラスファイルは, 膨大で継承しているだけでクライアントには不必要なファイルが多い. そこで, 送信されるオブジェクトの必要データだけを取り出して, クライアントに送信するようなデータの変換を行なっている. こうすることにより, 必要最小限の情報をクライアントに送信できるだけでなく, 不必要なクラスファイルをクライアントが持たなくてよいため, ファイル容量が節約できる.

3.4 提示のための制御機構

これまでの QUIK は記号によって記述されたオブジェクトのみを対象としてきた. そのため, 実行の順序はさほど重要ではなく, 記述された順に実行するものであった. しかし, 今回の機能拡張はマルチメディアを対象としているため, 以前のようにはいかない. 実行したい順に記述すれば良いのであるが, それでは, ユーザの負担が増え, 扱いにくいものと成り得ない. そこで実行順序を決定する上で 2 つの方法を用意した. それは, 逐次実行と非順序実行である. 逐次実行とは, ユーザの記述した順序のまま出力し, その順序は変更しない. 非順序実行とは, 順序に曖昧性がある場合などにこの方法を利用すると, 各メディアの種類を認識し, 順序を変更して出力するといったものである. 各メディアに重みを付け, それによって順序が決定される. その重み付けについては次で説明をする.

3.4.1 重み付け

実行順序を決定する方法として重み付けを採用した.

非順序実行の場合, 逐次順序実行のように実行する順序が明確に決定されていないため, 順序を変更するためのアルゴリズムが必要になる. 実行順序を決定するために重要なことの 1 つとしては, メディアの種類を認識して, その種類によって出力の順序を変更することがまず上げられる. つまり, 図 6 の 1) のように登場人物が出力されていないにも関わらず, その登場人物が会話を始める, などのような簡単な矛盾を解消することである. そのために各メディアが元々保持している自メディアを判別するための type の情報を利用し, それをそのまま重みとして再利用した. 出力の優先権は重みの小さな方から順に与えた, つまり, 2.1 章で示した順 (静止画, 動画, オブジェクト, テキスト, 移動, サウンド, 発話) に出力の優先権が与えられる.

重み付けを行う場合, メディア同士の場合, ルールのヘッド同士の場合あるいはメディアと

ルールのヘッドの場合の 3 種類に分けられるが, メディアのみでなくルールのヘッドにも重み付けより順序の変更をを行う必要がある. しかし, メディアのデータを保持していないルールのヘッドである場合 type の重みを保持していないためそれを利用することはできないため, 仮の内部データ構造にすべてのオブジェクトを変換して, そのデータに重みの情報を入れる変数を持たせた. メディアに重み付けをする場合は type をそのまま利用し, ヘッドに重み付けをする場合はヘッドが示すボディの重みの平均をヘッドの重みとした. この変数は type とは異なり, すべてのオブジェクトが平等に保持しているため, メディア同士の場合はもちろん, ヘッド同士やメディアとヘッド等の場合でも簡単に順序の変更が可能である.

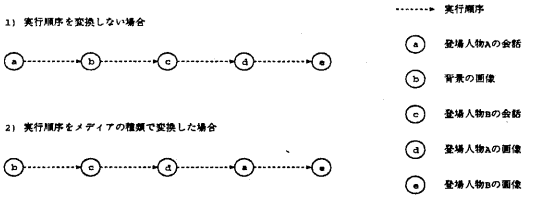

図 6: 重み付けによる実行順序

3.5 デバッガ

戯曲提示システムのデバッガは, テキストで書かれた戯曲からでは取り出せない情報を編集するためのものであり, 次の 3 つのウインドウによって構成されている.

シーングラフウインドウ
2.3 で説明した機能を持ったウィンドウを表示する.

詳細情報ウインドウ
シーングラフウインドウで示された現在実行しているノードの詳細情報を表示する. このデータを編集することによって, 詳細情報を決定していく (図 7(2)).

位置表示ウインドウ
通常 viewer は, 3 次元仮想空間からなる舞台を正面から見ているため, キャラクタなどの位置が分かりにくい. そこで位置表示ウインドウは, 舞台を真上から見たキャラクタの位置を示したウインドウである. 実行中のノードが Move の場合, このウインドウ上のキャラクタをマウスでドラッグさせることにより移動情報を入力することが可能になる (図 7(1)).

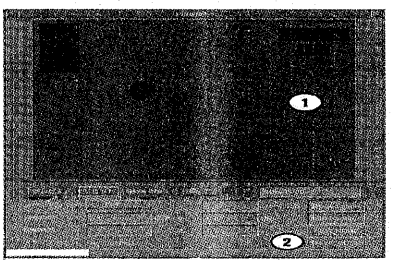

図 7: 詳細情報ウインドウ

3.6 通信機構

ダミークライアントの役割は, クライアントと Aglets サーバ間の通信をつかさどり, その全責任を負うものである. しかし, ただデータの橋渡しを行なうだけではなく次の 2 つの応用例を考えている.

3.6.1 ミラーリング

プレゼンテーションにおいて, その実行結果を複数の計算機上で表示させる事が出来る. そのためには戯曲を再生や編集している端末に対してミラーリング表示させたい端末を登録する必要がある. ただし, 結果の表示だけを許されており, 戯曲のコマンド操作および編集の権限は与えられない.

この機能により, 世界中の人に対してリアルタイムに戯曲を提示することができる.

3.6.2 クライアント間通信

これまでのQUIKシステムでは, クライアント間の通信はサポートされていなかったが, 今回はクライアント間での通信が可能となる. 文字だけでなくメディア情報などのオブジェクトもやり取りも可能である.

この機能と上記の機能を組み合わせることで, 戯曲を鑑賞しながら意見交換をしたり, 編集に対しての指示を出したりできる. さらには, メディア情報の交換することできる.

4 終りに

本論文では, 対象とする情報源を英数字などの記号で記述されたオブジェクトだけでなく, マルチメディアを扱うための方法について議論してきた. また, 新しい機能の応用としてテキストで書かれた戯曲を視覚化する方法を議論し, 実装してきた. しかし, 現在のQUIKでは, 以下の問題点がある.

今後の課題として, 通常の QUIK で導入されている仮説機能を, 戯曲のシナリオにおいても適用することが考えられる, これは「もし, この場面でこの人が出てきたら…」と言うシナリオに対して, それに対する戯曲の視聴覚化を行なうようにすると言ったものである.

また今回, デァドラ伝説の戯曲提示システムを作るにあたって, 登場人物の画像や建物などのオブジェクトなどのコンテンツがほとんどなかった. よって, コンテンツの整備も重要な課題と考えられる.

謝辞

さまざまな議論を頂く岡山県立大学横田研究室の皆様に感謝します. なお, 本研究の一部は文部省科学研究費基盤研究 (C) による.

参考文献

[1] 藤野猛士、野宮一生、横田一正、國島丈生、三宅忠明、"構造化文書に基づいた対話的戯曲提示システムの実現"、情報処理学会データベースシステム研究会、東京、2000年5月25-26日.

[2] Takeo Kunishima, Kazumasa Yokota, Bojiang Liu, and Tadaaki Miyake, "Towards Integrated Management of Heterogeneous Documents", *Cooperative Databases and Applications '99*, pp.39-51, Springer, Sep., 1999

[3] Bojiang Liu, Kazumasa Yokota, and Nobutaka Ogata, "Specific Features of the QUIK Mediator System", *IEICE Transactions on Information and Systems*, vol.82, no.1, pp.180-188, Jan., 1999.

[4] 三宅忠明、横田一正、"情報処理としての作品解析 — デァドラ伝説の研究から," 英語青年、vol.146,no.1, pp.6-9, April., 2000.

[5] 緒方啓孝、野宮一生、國島丈生、横田一正、"QUIKによる戯曲の視覚化", 電気・情報関連学会中国支部大会、122512, 1999年10月23日.

[6] 杉本健二、横川賢、緒方啓孝、國島丈生、横田一正、"QUIKシステムにおける分散探索の制御", 電気・情報関連学会中国支部大会、052517, pp.176-177, 1998年10月24日.

[7] 田中秀明、吉山雅彦、植村俊亮、"映像データベースのための同種メディアの統合", 情報処理学会データベースシステム研究会、114-5, pp.31-36, 1998年1月19日.

[8] Hoschka.P.(ed), "synchronized Multimedia Integration Language (SMIL) 1.0 Specification" (W3C Recommendation), 1998-06-15, http://www.w3.org/TR/1998/REC-smil-19980615.

[9] 林、折原、下田、上田、横山、八重樫、栗原、安村, "テレビ番組制作言語TVMLとその言語仕様", 映像情報メディア学会冬季大会、4-4, pp. 87, (1997)