プログラミング—言語・基礎・実践— 10−14
（１９９３．１．２９）

視覚的プログラミングによるGUIプログラムの生成（中間報告）

倉部 淳† 伊知地 宏‡

富士ゼロックス株式会社
†第一機能開発部
‡システム・コミュニケーション研究所

Abstract

GUIのダイアローグには制約を持つものが多い．しかし，制約を視覚的に記述する技術はまだ確立されているとは言えない．我々は，制約オブジェクト指向モデルを基に制約が論理関係で表せ，また幾何学的性質を持つことを導きだした．これをもとにダイアローグの制約を視覚的に記述することを考え，第一歩としてダイアローグの３種類の制約を論理型言語で記述し，それを視覚的に表現することを試みた．この制約は全て論理型言語で記述できたが，そのうちの２種類の制約は視覚的にも表現できた．

Program Synthesis for GUI by Visual Programming (Preliminary Report)

Jun Kurabe† Hiroshi Ichiji‡

† 1st Function Development Dept.
‡ Systems and Communications laboratory
Fuji Xerox Co., Ltd.

Abstract

Constraints in dialog is a very important part of GUI. But there are many problems to write constraints with visual programming. We find constraints have graphical chracteristics by the constraints object oriented model. Our goal is to describe constraints in dialog with visual programming. As first step to our goal, we tried to write three kind of constrains in dialog with Prolog, and show them as graphical relationships. All of constrains are written with Prolog, and two of them are transrated to graphical relationships.

1 はじめに

ソフトウェアの高機能化にともない使いやすいユーザインタフェースが求められてきている．グラフィカルユーザインタフェース (GUI) は，視覚に訴えるのでわかりやすいと言われ，色々なソフトウェアに使われはじめている．最近では，GUIのプログラムがアプリケーションプログラム全体のの 50% を越えることもある．GUIのプログラム開発の難しい点は，GUIのプログラムを走らせるまで，GUIが期待どおりにできたかがわからない点である．期待したGUIができなかった場合には，再びGUIの部分を作り直さなければならない．このため，GUIを確かめながら作成できる環境が求められている．

devGuide や Interface Builder，我々が開発した Preface[4] を用いると，シーハイムモデル [3] のプレゼンテーション部を視覚的に，ダイレクトマニュピレーションの操作で設計できる．さらに，GUIのソースコードも自動的に生成できる．これにより，ユーザインタフェースのプレゼンテーション部は視覚的にプログラミングできる．

しかしながら，これらのツールを用いても，GUIのすべてが実現できるわけではなく，GUIの制約や動作については既存の方法でプログラミングをしなければならない．制約は，オブジェクト間にある関係であり，ボタンが一列に整列することは制約である．また，ラジオボタンのように，二つのボタンが交互にONになるのも制約の一つである．このように制約はGUIにおいて重要であり，GUIの制約に関する研究も行なわれている．

Thinglab[1]では，表示画面上の図形の制約を視覚的に設定できる．例えば，2等辺三角形の2辺が等しいという制約を画面上で視覚的に定義できる．また，モデルの構造の制約を図形間の関係として表現し，視覚的に編集できるようにした結果もある．例えば、モデルの上下関係を図形の木構造の上下関係として表示し，編集ができる [2].

このようにプレゼンテーション部やモデルの制約を視覚的に定義することに関しては上で述べたようにいくつかの研究成果がでている．しかし，ダイアローグの制約を視覚的定義することに対し，色々なアプローチが試みられているが，技術的に確立されているとはいえない．我々は，ダイアローグの制約を視覚的に記述することを試みることにした．

以後，第二節で制約の幾何学的性質について述べ，第三節で制約を視覚的に定義する方法のアプローチと対象とする制約について述べる．第四節で，第三節で述べた制約を論理型言語で記述し，第五節で制約の視覚的表示について述べ，最後にまとめと今後の課題を述べる．

2 制約の幾何学的性質

以下では，ユーザインタフェースの要素をオブジェクトと考える．

定義 1 (近傍) Sを空間，$P(x,y)(x,y \in S)$ を述語とするときに，$x \in S$に対して x の近傍 $U(x,P)$ を以下のように定義する:

$$U(x,P) = \{y \mid y \in S, P(x,y)\}$$

ここで $P(x,x)$ が成り立ち，さらに $P(x,y)$ を満たす $y(\neq x)$ が少なくとも一つ存在するものとする．

定義 2 (近傍系) $x \in S$に対する近傍系 $V(x)$ を以下のように構成する:

1. $U(x,P) \in V(x)$ かつ $S \in V(x)$.
2. $U \in V(x)$ で $U \subset U'$ $(U' \in \mathcal{B}(S))$ ならば，$U' \in V(x)$. ただし，$\mathcal{B}(S)$ は Sの部分集合の全体を表す.
3. $U_1 \in V(x), U_2 \in V(x)$ ならば，$U_1 \cap U_2 \in V(x)$.
4. $V(x)$ の$\subset$に関する極小元が U ならば，任意の $y \in U$に対して $U \in V(y)$.

この近傍系 $V(x)$ でオブジェクト空間 Sに位相を入れる [7]. 次に制約を定義する．

定義 3 (制約) オブジェクト x と x のある近傍 $U(x,P)$ に属するオブジェクトとの間に成り立つ関係 Pを制約条件と呼び，オブジェクト x の近傍と制約条件の対 $(U(x,P),P)$ にオブジェクト x の制約という．

制約オブジェクト指向モデルは，上記のような位相空間上でオブジェクト指向をモデル化したものである．

定理 1 Sの空でない部分集合Oが開集合であるための必要十分条件は，Oの任意の点xに対して，Oがxの近傍となっていることである．

この定理を使っていくつか例を見てみよう．

例 1 最初にラジオボタンをモデル化した排他制御について考える．空間$S=\{a,b,c,d\}$とし，aとb，cとdがそれぞれ排他的であると考える．すなわち，以下が成り立つとする．

$$P(a,a),\ P(a,b),\ P(b,a),\ P(b,b),$$

$$Q(c,c),\ Q(c,d),\ Q(d,c),\ Q(d,d).$$

すると，各オブジェクトの近傍は以下のようになる．

$$U(a,P)=U(b,P)=\{a,b\},$$

$$U(c,Q)=U(d,Q)=\{c,d\}.$$

これから，近傍系を計算すると次の事実が得られる．

$$V(a)=\ V(b)=$$
$$\{\{a,b\},\{a,b,c\},\{a,b,d\},\{a,b,c,d\}\}$$

$$V(c)=\ V(d)=$$
$$\{\{c,d\},\{a,c,d\},\{b,c,d\},\{a,b,c,d\}\}$$

これより，Sの開集合族は

$$\{\emptyset,\{a,b\},\{c,d\},S\}$$

であり，閉集合が開集合の補集合であることから，閉集合族は

$$\{\emptyset,\{a,b\},\{c,d\},S\}$$

である．これよりすぐ分かるように，Sは$\{a,b\}$と$\{c,d\}$で開集合によって分割できるので連結でなく，またT_1空間でもない．この結果は2つの相異なる排他制御を持つ空間を如実に表している．すなわち，連結でないということはオブジェクトの共有がない全く異なる制約を持っていることを表しているし，T_1空間でないことはオブジェクトが密着した制約関係を持つことを表していると考えられる．

例 2 プリントツールをモデル化した例を考える．空間$S=\{a,b,c,d\}$とし，aがプリントボタン，bをプリントキュー，cとdがフィールドとする．

$$P(a,a),\ P(a,b),\ P(b,a),\ P(b,b),$$

$$Q(a,a),\ Q(a,c),\ Q(a,d),\ Q(c,a),$$

$$Q(c,c),\ Q(d,a),\ Q(d,d).$$

このときに，前と同様に計算を行なうと，Sが連結でT_1空間でないことが得られ．連結であることより全てのオブジェクトがお互いに影響し合い，T_1でないことよりオブジェクトが密着した制約が存在していることを示している．

例 3 空間$S=\{a,b\}$とし，aとbに制約関係があるとする．このときSは，密着空間になる．これより，全てのオブジェクトがお互いに影響し合い，非常に密着した制約が存在することがわかる．

例 4 空間$S=\{a,b,c\}$とし，それぞれのオブジェクトの間に相異なる制約関係があるとする．このときSは，離散位相空間になる．これより，全てのオブジェクトがお互いに影響を及ぼし合わず，離散的にしか制約が存在しないことになる．これは，aの変化が一回りして自分自身へ反映されることを示していると思われる．

制約の解消は，位相空間の間の同相写像と考えられるが，細かい考察はまだ行なっていない．

制約オブジェクト指向モデルでは，制約は二種類存在する．メソッドの実行で制約条件が成り立たなくなるときに実行を禁止する強い制約と，メソッドの実行を禁止しない弱い制約である．弱い制約では，制約条件が満たされなくなると自ら状態を変化させ，制約条件を成立させるようにする．制約オブジェクト指向におけるオブジェクトは，エージェントと考えることも出来る [6].

3 制約の視覚化にむけて

本節では，ダイアローグの制約を視覚的に記述するためのアプローチと対象とする制約について述べる．

3.1 制約の視覚化へのアプローチ

前節で述べたように，オブジェクト間の制約は述語で記述できる．すなわち，制約は論理関係である．さらに，制約オブジェクト指向モデルを用いると論理関係は幾何学的性質を持つ．よって，制約は幾何学的性質を持ち，視覚的に定義できる可能性が高い．

そこで，まず，我々は，制約を論理型言語 Prolog で記述し，その記述を視覚的に表現すること試みる．最終的には，視覚的プログラミング言語を作成し，制約を視覚的に記述して，論理プログラムの抽出を行ないたい．

3.2 実験対象

本論文では，3種類の制約，ラジオボタン，プリントキュー，プリントボタンについて，論理型言語での記述とその視覚的な表現を試みる．これらの制約は，ダイアローグによく現われる制約である．

ラジオボタンの制約は排他制御である．ラジオボタンは，2個のボタンで構成され，それぞれが2つの値を取る属性を持つ．ボタンが互いに異なる属性値をとることが制約条件である．一方のボタンの属性値が変わると制約の解消が起こり，他方のボタンの属性値を変え制約条件を成立させる．

プリントキューは，長さ1のキューとキューの入力の間の制約である．キューは Empty と notEmpty の値をとる属性を持ち，キューの入力は on と off の値をとる属性を持つ．制約条件は，キューの属性値が empty ならば入力の属性値は on の状態であり，キューの属性値が notEmpty だとキューの入力の属性値は off である．キューの属性値が empty から notEmpty に変わると，制約の解消によりキューの入力の属性値は on から off になる．逆も同様である．

プリントボタンは，二つの入力フィールドとの間に制約である．プリントボタンは二つの属性 a, b を持ち，どちらも on, off の値をとる．フィールドはそれぞれ empty と notEmpty の値をとる属性を持つ．両方の入力フィールドの値が notEmpty ならば属性 a, または b のどちらか一方が on の値をとる．さもなければ off の値をとる．

4 制約の Prolog による記述

本節では，第二節で述べた制約を論理型言語 Prolog を用いて記述する．

4.1 ラジオボタン

ラジオボタンを構成する二つのボタンを a, b とする．ボタンのモデルにおいて，button はオブジェクト名と属性とを結び付ける．

```
button(a, on).
button(b, off).
```

ボタン a, b の間にある制約は双方向の関係で，二つの式で表される．

```
relationship(a, b).
relationship(b, a).
```

ボタンの初期化の制約は，

```
init(X, Y) :- relationship(X, Y),
              button(X, on),
              button(Y, off).
init(a, b).
```

である．

制約解消プログラムは，ボタン a, b の間で値を交換することであり，

```
change(X, Y) :- relationship(X, Y),
                retract(button(X, V)),
                retract(button(Y, W)),
                assert(button(X, W)),
                assert(button(Y, V)).
```

と書ける．

4.2 プリントキュー

プリントボタンを a, プリントキューを b とする．プリントボタン, プリントキューはそれぞれ属性を一つ持ち，初期値は on と empty である．

```
button(a, on).
queue(b, empty).
```

プリントボタンとプリントキューは，双方向の制約があるので，二つの式で記述できる．

```
relationship(a, b).
relationship(b, a).
```

初期状態は，以下のように記述できる．

```
init(X, Y) :- relationship(X, Y),
              button(X, on),
              queue(Y, empty).
init(a,b).
```

制約 constraints は

```
constraints(X, on) :-
        relationship(X, Y),
        queue(Y, empty).
constraints(X, off) :-
        relationship(X, Y),
        queue(Y, notEmpty).
```

である.

ボタンが押されたときの制約解消プログラムが down であり，キューが empty になったときの制約解消プログラムが finish である.

```
down(X, on) :-
         relationship(X, Y),
         retract(queue(Y, empty)),
         assert(queue(Y, notEmpty)),
         constraints(X, V),
         retract(button(X, off)),
         assert(button(X, V)).
finish(X, off) :-
         relationship(X, Y),
         retract(queue(Y, notEmpty)),
         assert(queue(Y, enmpty)),
         constraints(X, V),
         retract(button(X, on)),
         assert(button(X, V)).
```

4.3 プリントボタン

プリントボタンを a とし，フィールドをそれぞれ b, c とする. プリントボタンは属性を二つ持ち，それぞれの初期値は off と off であり，フィールドはそれぞれ属性をもち初期値は empty である.

```
button(a, off, off).
field(b, empty).
field(c, empty).
```

プリントボタンとフィールド b, c とは一方向の関係 relationship があり，また，フィールド b, c 間には双方向の関係 family がある. ボタン a とフィールド b, c の間にも group の関係がある.

```
relationship(a, b).
relationship(a, c).
family(b, c).
family(c, b).
group(X, Y. Z) :- family(Y, Z),
                  relationship(X, Y),
                  relationship(X, Z).
```

プリントボタン a，フィールド b, c の 3 個のオブジェクトの制約 constraints は，以下の 4 つの式で記述される.

```
constraints(X,  off, off) :-
               group(X, Y, Z),
               field(Y, empty).
constraints(X, off, off) :-
               group(X, Y, Z),
               field(Z, empty).
constraints(X, on, off) :-
               group(X, Y, Z),
               field(Y, notEmpty),
               field(Z, notEmpty).
constraints(X, off, on) :-
               group(X, Y, Z),
               field(Y, notEmpty),
               field(Z, notEmpty).
```

制約解消プログラムは，まず, 実行可能性を調べ, 二番目の属性を設定する. 次に二番目の属性値が on ならば, 三番目の属性値を on すると同時に二番目の属性値を off にしている.

```
change(X) :-  group(X, Y, Z),
              constraints(X, off , V),
              retract(button(X, off, _)),
              assert(button(X, off, V)),
              button(X, off, on),
              retract(button(X, A, B)),
              assert(button(X, B, A)),.
```

5 制約の視覚的記述

本節では，Prolog を用いて記述した制約を視覚的に表現する.

5.1 ラジオボタン

ボタンは，オブジェクトと属性の組として図 1 のように表現できる. 線の矢印は，関係の方向を示

している．関係 relationship は，図 2のようにオブジェクト間に relationship のタグのついた線で表す．relationship は双方向の関係であるので，二本の線で表現される．

しかし，init，change の視覚的表現については検討中である．

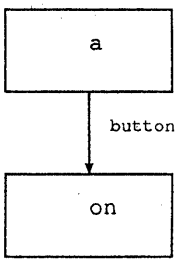

図 1: button

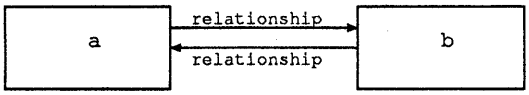

図 2: relationship

5.2 プリントキュー

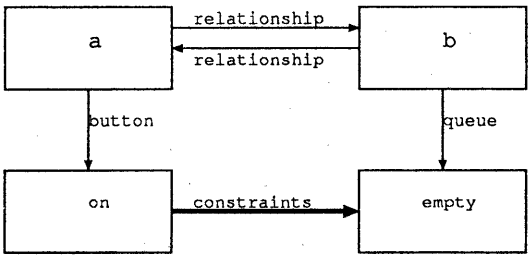

図 3: constraints

プリントボタン，キューのは，ラジオボタンとは属性の違いだけであるので，同様に表現できる．さらに，relationship は，ラジオボタンの relationship と全く同じに表現できる．constraints は，属性の関係として表現し，図に示す．ただし，この表現だとすべての制約条件を同時に表示することはできない．

finish, down の制約解消プログラムを視覚的に表現する方法は今後の課題である．

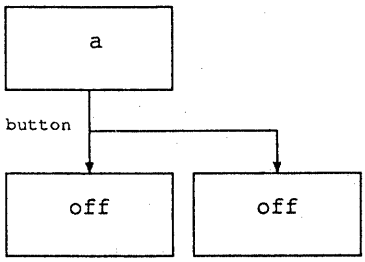

図 4: print button

5.3 プリントボタン

プリントボタンは，属性を二つ持ち図 4のように表現される．フィールドは，ラジオボタンと同様に，また，関係 relationship, family もラジオボタンの relationship と同様に表現できる．しかし，制約 constraints, 制約解消 change および group の表現方法は今後の課題である．

6 まとめと今後の課題

GUI のダイアローグの制約から，3 種類の制約，ラジオボタン，プリントキュー，プリントボタンを論理型言語 Prolog で記述し，さらに，Prolog で記述された制約を視覚的に表現する方法について述べた．

ここで，対象とした 3 種類の制約は，どれも論理型言語 Prolog を用いて記述できた．視覚的な表現については，個々のオブジェクトや 2 つのオブジェクト間の関係については表現できた．しかし，3 個以上オブジェクト間の関係や制約，制約解消プログラムを視覚的に表現できていない．これの検討を今後進める予定である．

ダイアローグには，この 3 種類以外にも色々な制約がある．それらの制約にも視覚的な表現の可能性，さらに視覚的にプログラミングする可能性を探るとともに，視覚的な表現手段の実装も行なう予定である。

参考文献

[1] Borning, A.: Defining Constraints Graphically, CHI'86 Proceedings, P.137-143.

[2] Takahashi, S., Matsuoka, S., Yonezawa, A.: A General Framework for Bi-Directional Trans-

lation between Abstract and Pictorial Data, UIST'91, pp.165-174.

[3] 萩谷昌己他: ウィンター・チュートリアル「ヒューマンインタフェースの最先端」, 日本ソフトウェア科学会 (1990).

[4] 伊知地宏, 倉部淳:ユーザインタフェース開発ツールの構成と実現, 情報処理学会ヒューマンインタフェース研究会, HI-37-3 (1991).

[5] 倉部淳, 伊知地宏: GUI のプログラム生成はどこまで可能か, 情報処理学会夏のプログラミング・シンポジウム.

[6] 中島秀之: エージェントモデル, コンピュータソフトウェア, Vol.9, No.5 (1992), pp.3-11.

[7] 松坂和夫: 集合・位相入門, 岩波書店, 1968.