制約指向に基づくオフィス業務記述法の分析 *

2R－8

石井 恵　金田 重郎
ＮＴＴコミュニケーション科学研究所

1 はじめに

著者等は，保守性が高く，ユーザに使い易いオフィスシステムの実現を目指した研究を行なっている．そして，手続き的な処理として考えられがちなオフィス処理を，整合性管理処理としてモデル化し，データ間の関係を制約として表現することにより，事務処理を実現する手法を提案した [1]．本手法は，(1) 制約充足エンジンにより，与えられた制約 (業務知識) に整合する処理結果を生成するため，処理結果の整合性を保証し，また，少ないプログラム記述量でシステムの実現が可能，(2) 制約関係を中心に業務を分析するので，例外処理の洩れの防止効果大，という特徴をもつ．

上記手法において重要なことは，事務処理に存在する制約関係の抽出，及び，その制約表現法である．本稿では，実システムである総務業務エキスパートシステム KOA[2] のタスクに対する，制約を主体とした業務分析，及び，その制約表現法について報告する．

2 総務業務エキスパートシステム KOA

KOA は，各種申請帳票の作成を支援する社内システムであり，データエントリシステムの一種である．帳票数は 33 種，データ項目数は，およそ 900 個である．KOA は，社員の結婚等のイベントに対し，社員から対話形式でデータを取得し，社内，共済組合等に提出する帳票を作成する．帳票は，社員に起こった出来事 (イベント) と，事務規則に基づき，決定される．KOA では，複数の業務に跨るデータを扱っているため，社員に起こったイベントの各業務への洩れのない通知が必要である．

3 制約表現のための業務分析

イベントに対して，正しく帳票を作成するためには，帳票間に存在する制約関係を洗い出さなければならない．そこで，帳票の種類を，(1) 作成されるタイミング，及び，(2) 各帳票上に存在するデータ項目の種類の観点から分析した (表 1)．なお，KOA が扱うデータには，帳票以外に各社員のデータ，社員の家族のデータが存在する．

(1) 作成タイミング

作成タイミングにより，帳票は以下の 4 種類に分類できる．

独立型: 他のイベントに関係をもたないイベントに対して作成される．

差分型: 従来の届け出値に変更があった場合に提出する必要がある．逆に，当該帳票が提出された場合には，従来の届け出値に必ず変更があったことを意味するので，関連したデータ項目は，必ず変更されなければならない．

差分＋希望型: 作成の前提として，ある条件を満たすことと，本人の希望が必要な帳票である．例えば，単身赴任は，本人と家族 (配偶者) の住所が異なることが必須であるが，本人が希望しなければ，単身赴任とならない．

View 型: データベースの View としての役割をもつ帳票であり，データベース中のデータと共に常に存在する．

(2) データ項目

データ項目により，帳票は以下の 2 種類に分類できる．

表 1: 帳票の分析結果

帳票を構成するデータ項目		作成タイミング: 独立型	差分型	差分＋希望型	View
	単一型	・出産休暇願 ・転出者調書 ・組合休暇願 ・社員証亡失届 ・社員記章亡失届 ・社員記章複数交付申請書 ・カード再交付申請書 ・ECL名札交付願 ・組合員証再交付申請書 ・駐車許可願（新規） ・仮居住地届 ・仮居住地確在証明書 ・振行振込依頼書 ・休職（復職）願 ・社宅入転居申込書	・健保住所変更届 ・出産費請求書 ・持ち家手当申請書 ・住民手当申請書（借家） ・婚姻届 ・改氏名届 ・弔慰金請求書 ・駐車許可願（変更） ・社宅入退居兼社宅 ・使用料控除依頼書 ・埋葬料請求書	・結婚休暇願 ・単身赴任届 ・遠隔地費扶養者交付申請書	
	集約型 単一型可		・扶養親族認定申請書（給与） ・被扶養者申請書（健保） ・氏名変更届（健保） ・急引休暇願		
	集約型 単一型不可				・社員連絡調書

単一型: 帳票を構成するデータ項目数が一定の帳票である．例えば，休暇届け等は，社員の氏名，所属，取得年月日等，記入すべきデータ項目は常に一定数である．

集約型: いくつかのデータの集合を含み，状況によって記入項目数が異なる帳票である．例えば，連絡調書では，扶養家族のみを記入する欄が存在する．この場合，扶養の家族の数によって，記入されるデータ数は異なる．但し，現状では集約型となっている KOA の帳票の中には，単一型に変更できるものも多い．例えば，家族の健康保険証の氏名変更届けは，現状では，1 枚の帳票で，氏名変更する家族を列挙する集約型であるが，氏名変更というイベントは，家族各々別々に起こる場合もあるので，改氏名する各家族毎に帳票を作成すべきである．

4 制約シンタックス

本シンタックスでは，ユニットと呼ばれる構造をもたないデータ構造を扱う．一方，通常，事務処理上のデータでは，帳票や社員データ，家族データといった，一塊の構造を持つ．よって，構造のあるデータから構造をもたないデータの変換が必要となる．この変換については，次章で述べる．以下，使用する制約シンタックスについて簡単に説明する．

ユニット: 各ユニットの値を定義する．値は定数，または，変数である．値が変数の場合，ユニットには，特定の値が割り付けられていないことを示す．

```
unit(ユニット名,　値).
```

制約: 制約は，次の element で定義される 1 つ以上の制約素から構成される．制約名はユニークである．制約条件を重複なく記述するため，制約素相互に以下の関係を設ける．

[制約素の排他性] 同一制約内の制約素は，あるユニット状態において，1 つの制約素が満足される場合，残りの制約素は必ず，そのユニット状態に矛盾しなければならない．

```
constraint(制約名,　(制約素名 1,　制約素名 2,　…).
```

制約素: 許されるデータの状態を定義する．制約素名はユニークである．pattern は各ユニットのとるべき値を表す．値には，定数，または，制約変数を書く．TEST は，ユニットが満たすべき Boolean 関数条件を表す．TEST で評価される関数の引数は，定数，または，その TEST と同じ制約素内の制約変数である．

---

*Constraint oriented approach for office systems
Megumi ISHII and Shigeo KANEDA
NTT Communication Science Laboratories

element(制約素名, (pattern(ユニット名1, 値1), ···,
TEST(Boolean 関数1), ···).

5 制約記述法

5.1 データ構造の変換

帳票，社員データ等，フレーム構造をもつデータの，ユニットへの変換について述べる．各フレームのスロットは，各々1つのユニットに対応付けられる．各フレームの存在(各帳票，社員データ等がそれぞれ存在するか否か)を示すユニットを作成する．また，データの変化を検出するため，差分型帳票の作成タイミングを規定するデータ項目に対しては，変化前と変化後に対応するユニットを作成する．

ここで問題になるのは，集約型の帳票に対するデータ構造である．集約型の帳票では，通常，可変個のデータの塊を記入するために，複数の空白欄から構成される空白行が用意されている．今回は，各行の各空白欄に対して1つずつユニットを割り付けた．

5.2 帳票間の制約

帳票に関する制約としては，(1)帳票の作成タイミングを規定する制約と，(2)帳票を構成するデータ項目間の制約の2種がある．以下の制約記述では，ある帳票が存在しないことを示す制約素では，その帳票を構成するデータ項目を，その制約素で参照しないこととした．これは，存在しない帳票に関しては，その上への値の書き込みの有無は無関係である，という思想に基づく．各帳票タイプに対する，本シンタックスを用いた制約を示す．

5.2.1 作成タイミングを規定する制約(作成判断)

独立型: 基本的には，その帳票が有る場合と無い場合を1つづつ宣言すれば良い．但し，KOAでは，社員データの存在と連動するため，以下の3種類の制約素が必要である．

```
constraint(特別休暇制約, (C1-1, C1-2, C1-3)).
element(C1-1,(pattern(特別休暇願い, 有),
              pattern(社員データ, 有)).
element(C1-2,(pattern(特別休暇願い, 無),
              pattern(社員データ, 有)).
element(C1-3,(pattern(特別休暇願い, 無),
              pattern(社員データ, 無)).
```

差分型: 値の変化を表す条件記述を含む．

```
constraint(改氏名制約, (C2-1, C2-2, C2-3)).
element(C2-1,(pattern(改氏名届け, 有),
              pattern(社員データ, 有),
              pattern(社員データの昔の氏名, ?X),
              pattern(社員データの今の氏名, ?Y),
              TEST(?X≠?Y)).
element(C2-2,(pattern(改氏名届け, 無),
              pattern(社員データ, 有),
              pattern(社員データの昔の氏名, ?X),
              pattern(社員データの今の氏名, ?X)).
element(C2-3,(pattern(改氏名届け有無, 無),
              pattern(社員データの有無, 無)).
```

差分＋希望型: 値の変化を表す条件記述，及び，希望の有無を表すユニットの値を含む．

```
constraint(単身赴任制約, (C3-1, C3-2, C3-3, C3-4, C3-5)).
element(C3-1,(pattern(単身赴任願い, 有),
              pattern(社員データ, 有),
              pattern(家族データ, 有),
              pattern(社員データの社員の住所, ?X1),
              pattern(社員データの社員の旧住所, ?X2),
              pattern(家族データの家族の住所, ?Y1),
              pattern(家族データの家族の旧住所, ?Y2),
              TEST((?X1≠?X2 ∨ ?Y1≠?Y2) ∧ ?X1≠?Y1),
              pattern(社員データの社員の単身赴任希望, 有)).
element(C3-2,(pattern(単身赴任願い, 無),
              pattern(社員データの, 有),
              pattern(家族データの, 有),
              pattern(社員データの社員の住所, ?X1),
              pattern(社員データの社員の旧住所, ?X2),
              pattern(家族データの家族の住所, ?Y1),
              pattern(家族データの家族の旧住所, ?Y2),
              TEST((?X1≠?X2 ∨ ?Y1≠?Y2) ∧ ?X1≠?Y1),
              pattern(社員データの社員の単身赴任希望, 無)).
element(C3-3,(pattern(単身赴任願い, 無),
              pattern(社員データ, 有),
              pattern(家族データ, 有),
              pattern(社員データの社員の住所, ?X),
              pattern(家族データの家族の住所, ?X),
              pattern(社員データの社員の単身赴任希望, 無)).
element(C3-4,(pattern(単身赴任願い, 無),
              pattern(家族データ, 無),
              pattern(社員データ, 有)).
element(C3-5,(pattern(単身赴任願い, 無),
              pattern(社員データ, 無)).
```

View 型: 帳票が参照するデータが存在する場合と存在しない場合について，1つづつ記述する．

```
constraint(連絡調書制約, (C4-1, C4-2)).
element(C4-1,(pattern(連絡調書, 有),
              pattern(社員データ, 有)).
element(C4-2,(pattern(連絡調書, 無),
              pattern(社員データ, 無)).
```

5.2.2 帳票を構成するデータ項目間の制約(転記)

同一帳票内の制約，および，他の帳票上，社員データ等との値の関係を宣言する．

単一型: 作成タイミングが独立型・View 型の帳票に関しては，作成タイミングとデータ間の関係を1つの制約で容易に表現できる．一方，作成タイミングが差分，差分＋希望型の帳票に関しては，作成タイミングに関する制約とデータ項目間の制約は，別の制約として記述した方が可読性が良い．著者等が提案した，制約充足エンジンは unification によるパターンマッチを行なうため，下記(1)，(2)のように同じ変数を共有するユニット相互は，値の伝播が可能である．そのため，値が等しいという表現は，極力，変数の共有により記述すべきである．unification による値の伝播は，事務処理上でよく行なわれる，データの転記処理を実現する．

```
constraint(特別休暇項目制約, (C5-1, C5-2, C5-3)).
element(C5-1,(pattern(特別休暇願い, 有),
              pattern(社員データ, 有),
              pattern(社員データの社員氏名, ?X),—(1)
              pattern(特別休暇願いの社員氏名, ?X),—(2)
              pattern(特別休暇願いの提出理由, ?Y)).
element(C5-2,(pattern(特別休暇願い, 無),
              pattern(社員データ, 有)).
element(C5-3,(pattern(特別休暇願い, 無),
              pattern(社員データ, 無)).
```

集約型: 可変部分に対しては，複数空白欄から構成される空白行の各行を，各データの塊に対応させ，各々制約とする．なお，KOAの帳票で集約型は1つのみなので，本稿では，制約表現は省略する．

6 まとめ

本稿では，総務業務エキスパートシステムKOAのタスクを例とし，事務処理中に存在する制約の分析，及び表現法について述べた．今後は，様々な事務処理に対して制約の分析を行ない，事務処理中に存在する制約を体系化していく予定である．

参考文献

[1] M.Ishii, Y.Sasaki and S.Kaneda: Interactive Constraint Satisfaction for Office Systems, *Proc. of the 10th IEEE Conference on Artificial Intelligence for Applications*,pp.116-124(1994).

[2] 中野: 容易に知識修正ができるオフィス業務エキスパートシステム，電子情報通信学会，人工知能と知識処理研究会，AI91-80,pp.49-56(1992).