LOTOSの抽象データタイプ実装のための支援法に関する一考察

5P−5 A Study on Implementation Support Method of LOTOS ADT

Pairoj TERMSINSUWAN, Zixue CHENG, Norio SHIRATORI

Faculty of Engineering, Tohoku University

1. INTRODUCTION

LOTOS[1] is an FDT developed within ISO for formal specification of communication software. ADT(Abstarct Data Type) is a part of LOTOS that deals with the formal description of data. So far, due to the high abstractness of ADT, problem in implementation is that there are too many possible concrete datas for one ADT and no gurantee whether all of them are appropriate for practical application. Decision for appropriate chioces depends on many factors for instance, application field, type of system etc. and a lot of intellectual works from implementer are necessary to solve this problem.

The main purpose of this paper is to present an implementation support method to reduce these intellectual works.

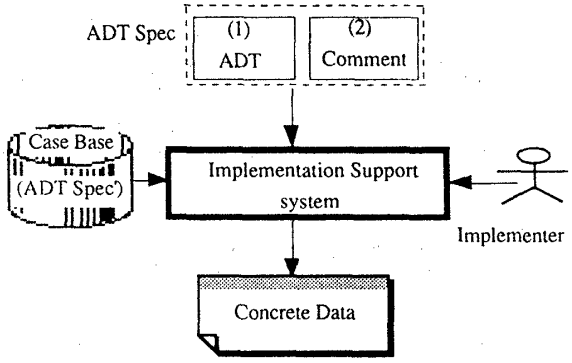


Fig.1 outline of the implementation support

2. OUTLINE OF THE SUPPORT METHOD

Our approach is to design a support method that presents concrete data of similar ADT specification as a candidate in implementation of new ADT specification. In addition to this, our support can acquire the comments which shows intentions from specifier as a factor to decide the most appropiate candidates in implementation. Fig.1 shows outline of the support. The implementation support retrieves ADT along with the comments part, compares it with the existing ADT specifications in case-base and outputs concrete data of specification that is most similar to input specification. Implementer modifies this concrete data to get final implementation.

A Study on Implementation Support of LOTOS ADT
Pairoj TERMSINSUWAN, Zixue CHENG,
Norio SHIRATORI
Faculty of Engineering,Tohoku University

3.SIMILARITY OF ADT SPECIFICATION

This is the most important part in design of the support. To obtain the most similar ADT in comparison , two kinds of similarities are neccessary

(1) Similarity of ADT, for comparison from Abstarct point of view

(2) Similarity of Comment,for comparison from specifier's intention point of view

3-1 Similarity of ADT

We have proposed the idea of ADT Specification Model(ADTM)[3] by further classifying Operations, especially constructors part[2] into four patterns (Fig.2).Similarity of ADT is defined based on this model. Any constructors part(Ωc) can be considered as the combination of these following patterns

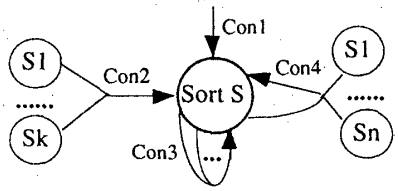


Fig.2 Four patterns of constructor

①*Primitive Constructor* defines a constant element

Pattern1 Con1: --> *S*

where Con1 is the constructor name

S is the range sort of data

②*Parametric Constructor* defines an element from *k* elements of domain sort S1,S2,..Sk.

Pattern2 Con2: S1,S2,..Sk--> *S*

where $Si, 1 \leq i \leq k$ is the *i* th domain sort

③*Recursive Constructor* defines a new element recursively from *k* existing elements (of sort① or ②) (*k* times)

Pattern3 Con3: *S,S,..S* --> *S*

④*Recurive constructor with additional input* creates a new element recursively from *j* exisiting elements(of sort① or ②) in the data of sort*S* and *k* elements from other *k* domain sorts.

Pattern4 Con4: *S,S,..S,S1,S2,..,Sk* --> *S*

(*j* times)

Similarity comparison of ADT1, ADT'1 with constructors part Ωc, $\Omega' c$ respecitvely is defined as

$$S_{ADT} = \frac{\sum_{i=1}^{4} Sc_i}{N}$$,where

N = number of different patterns in both ADTs

Sci = similarity of pattern i constructor, where

$$Sc_1 = \frac{N_1}{N'_1}$$

$$Sc_2 = (\frac{N_2(1)}{N'_2(1)} + \frac{N_2(2)}{N'_2(2)} + ... + \frac{N_2(k)}{N'_2(k)}) /k$$

$$Sc_3 = (\frac{N_3(1)}{N'_3(1)} + \frac{N_3(2)}{N'_3(2)} + ... + \frac{N_3(k)}{N'_3(k)}) /k$$

$$Sc_4 = (\frac{N_4(1,1)}{N'_4(1,1)} + \frac{N_3(1,2)}{N'_3(1,2)} + ... + \frac{N_3(j,k)}{N'_3(j,k)}) /(j*k)$$

where

N1, N'1 = numbers of con1 in ADT1, ADT'1

N2(k), N'2(k) = numbers of con2 with k domain sort in ADT1, ADT'1

N3(k), N'3(k) = numbers of con3 with k domain sort in ADT1, ADT'1

N4(j, k), N'4(j, k) = numbers of con4 with i domain sorts and other k domain sorts in ADT1, ADT'1

3-2 Similarity of comment

Comment is the logical expression retrieved from Specifier and kept in system as an internal expression(IE) which is a part of system knowledge. Representation of system knowledge is a m-tuples of trees C1,...,Cj,...,Cm. Each Cj is knowledge on one specific field called "category" which consists of a set of keywords and relations between keywords (Fig.3).IE is a m-tuple of specific path (Pj) in each Cj selected by specifier's intention and is defined as

IE = (P1, P2,...,Pj,...,Pm)

where Pj = (ϑ1,...,ϑi,...,ϑf)

ϑi is the ith keyword in Cj

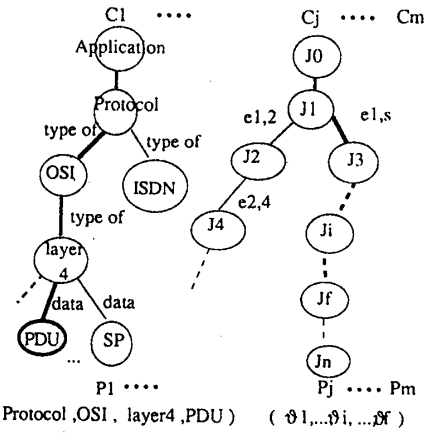


Fig.3 System knowledge and Internal expression

Similarity comparison of comment1, comment1' represented as IE, IE' is defined as

$$S_{com} = \frac{\sum_{j=1}^{m} S_{cat}(Pj,P'j)}{m}$$, where

Scat(Pj,P'j) is similarity of comment in the jth category.

$$S_{cat}(Pj,P'j) = \frac{\sum_{i=1}^{f} M(\vartheta i, \vartheta' i)}{Nj}$$, where

Nj = numbers of differnt keywords in Pj,P'j

$$M(\vartheta i, \vartheta' i) = 1 \text{ if } \vartheta i = \vartheta' i$$

else = 0

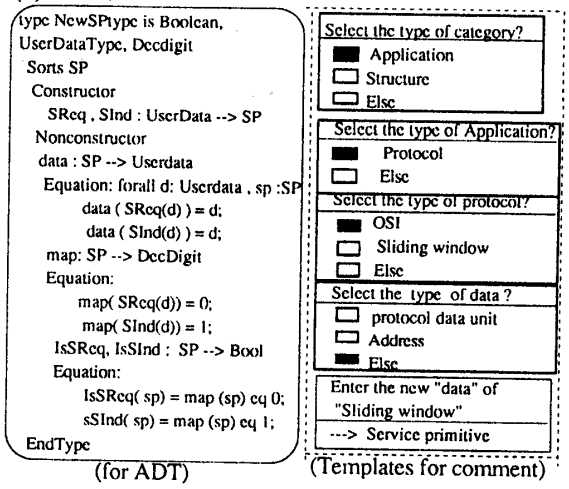
4. APPLICATION EXAMPLE

This example shows how our method works

(1) System retrieves ADT along with specifier's intention by interface which are a set of templates for ADT and comment in Fig.4(a)

(2) System retrieves the most similar ADT by similarity comparison and outputs concrete data of this ADT in Fig.4(b) as a candidate for implementor to modify. The result of modification is in Fig.4(c)

(a) interface



(b) output

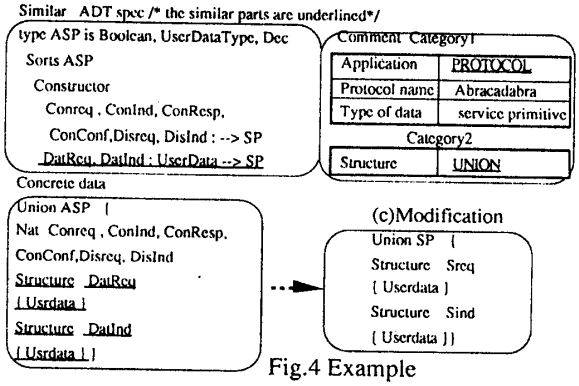


Fig.4 Example

5. CONCLUSION

We have proposed design of implementation support for ADT. Implementation of this support is now under-developing

REFERENCE

[1] LOTOS -A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour,"ISO8807,

[2] ISO, "Contribution on LOTOS enhancements" ISO/IEC JTC 1/SC21 ,April 1992

[3] Pairoj TERMSINSUWAN, Zixue CHENG, Norio SHIRATORI, "A Support Environment for Implementation of FDT Specifications", Technical Report of IEICE, IN92-38