A. Malioukov

NON OBJECT-BASED APPROACH TO THE B FORMAL METHOD

1 Introduction

The work presents the B specification and implementation of ACVIS. B is a method for specifying, designing, and coding software systems [1]. ACVIS is A Computerized Visitor Information System. This system assists in managing arrangements for visitors and meetings at a large site.

2 The B Method

Application of formal methods for software engineering is an important issue these days. The formal methods are used in academic research as well as in industrial applications. One of the most promising formalisms is the B-Method. The B-Method is a model-oriented method used in industry to develop safety critical software systems. The B-Method and the Abstract Machine Notation, often abbreviated to AMN, were developed by J-R. Abrial from 1980 onward. AMN is a combination of mathematics and substitution that is used in specification and design in the B-Method [5]. The mathematical framework was provided by using predicate transformers and by extending E.W. Dijkstra's Weakest Precondition Calculus [7]. Work by R-J. Back, J.M. Morris and C.C. Morgan and others had illustrated how Dijkstra's calculus for programs could be extended to cover abstract specification. These extensions inspired the development of B's Generalised Substitution Language (GSL) for use within the B-Method. GSL adds the predicate transformers of pre-conditions and unbounded non-determinism to Dijkstra's notation for guarded commands. During these three years, J-R. Abrial devised most of the features specific to AMN and the B-Method. Discussion with C.C. Morgan and the collective work at the Programming Research Group (PRG) at Oxford University have had a significant influence on this work.

The B method is close to VDM (Vienna Development Method) [12] and to Z [13]. Obviously, many ideas of both these methods can be recognized in B. Z is the result of an effort started by J-R. Abrial around 1975 in Grenoble, and continued in Paris and Oxford. Abrial shared an office with C.B. Jones [11] during the same period. From him J-R. Abrial learned the idea of program development and the concept of refinement and its practical application, under the form of proof obligations [1].

There are two toolkits supporting the B formalism. The first one, the B-Toolkit [2], was released by B-Core (UK) in 1994. The second one, Atelier B [3] has been produced by Steria (Digilog) in collaboration with J-R. Abrial. Both tools have similar facilities and provide support for all development process, syntax, type and static semantics checking, animation, proof obligation generation, automatic and interactive proof, code generation and pretty-printing (in LaTeX and HTML formats). The first significant industrial success in formal development has been achieved in France, where the safety critical software of the METEOR system (a new line of metro in Paris) was formally developed with the help of Atelier B by Matra Transport International [15].

The International B Conference Steering Committee (Association de Pilotage des Conferences B - APCB) [8] has as its purpose to support, further and develop the study of the B Method for specification and formal development of software. There is a user group, called the BUG [9], for exchanging and discussing information on B. Every year APCB organizes the International B Conference, supported by the BUG and the ZUG (the Z User Group) [10].

83
3 Problem Description

We illustrate our approach by the B specification of a Computerized Visitor Information System (ACVIS). ACVIS based on a Computer Aided Visitor Information And Retrieval system (CAVIAR). A formal specification of CAVIAR was given in 1981 by J-R. Abrial and was subsequently specified in Z by B. Flinn and I.H. Sørensen [6]. In 1992, R. Docherty and A. Diller presented CAVIAR in AMN [14].

The following information is initially given: Visitors come to the site to attend meetings. A visitor may require a hotel reservation. Each meeting is required to take place in a conference room. A meeting may require the use of a dining room for lunch. Booking a dining room requires lunch information, including the number of places needed.

Some important properties:
1. At any time a conference room may be associated with only one meeting.
2. At any time a meeting may be associated with more than one conference room.
3. At any time a meeting may be associated with only one dining room.
4. At any time participants from several meetings can occupy the same dining room. Each dining room has a maximum capacity and unnumbered seats.
5. At any time a visitor may be associated with only one meeting.
6. At any time a meeting may involve several visitors.
7. At any time a hotel room may be associated with only one visitor.
Remark: We omit the notion of time but it can be added, if needed.

4 Non Object-Based Specification of the ACVIS

The B-Method is a formal model oriented method based on set theory and relies on an extention of the weakest precondition calculus. A program derivation in the B includes such steps as a specification, a number of refinements and an implementation which are represented as abstract machines. The specification is a formal description of a problem domain. By the refinement steps specification can be transformed into the implementation. The implementation can be automatically translated into C-code and it is not allowed to contain any nondeterminism which is usually involves on the specification step.

Abstract Machine Notation (AMN) is using for specifying the operations. Statement in AMN can be a simple substitution (~x := e), a .scip-substitution, a precondition substitution (PRE P THEN S END), and/or guarded substitution (SELECT P THEN S END), where P is a predicate, S is a substitution, a; e is a variable and e is an expression. Each substitution defined as a predicate transformer which transforms a postcondition Q into the weakest precondition for S to establish Q, wp(S, Q).

The structure of AMN specification will be described on the above example of non object-based specification of the ACVIS. The specification is identified by the unique machine name ACVIS. Dining-Room-Capacity is an input parameter, which should be type of natural numbers or sets. The set of constraints on the parameters usually presented in the CONSTRAINTS clause. In our example, we do not have such clause, because Dining-Room-Capacity is a natural number. CONSTANTS clause introduces constants and PROPERTIES clause gives their definitions.

MACHINE ACVIS (Dining_Room-Capacity)
VARIABLES
register'-book'-for'-visitors, hotel-rooms-booking, register'-book-for'-meetings, meeting-visits, conference-rooms-booking, dining-rooms-booking
IN Variant

register-book-for-visitors C PEOPLE \[\]

hotel-rooms-booking is a set of partial injections from each registered visitor to one hotel room from the set HOTEL-ROOMS (also known as 'one-to-one' relations).

hotel-rooms-booking G register-book-for-visitors \[\] HOTEL-ROOMS \[\]
register-book-for-meetings C MEETINGS \[\]

meeting-visits is a set of partial functions from some registered visitor to some registered meetings (also known as 'one-to-many' relations).

meeting-visits C register-book-for-visitors \[\] register-book-for-meetings \[\]

conference-rooms-booking is a set of partial functions from some conference room from the set CONFERENCE-ROOMS to some registered meetings.

conference-rooms-booking \[\] CONFERENCE-ROOMS \[\] register-book-for-meetings \[\]

dining-rooms-booking is a set of partial functions from some registered meeting to some dining rooms from the set DINING-ROOMS.

dining-rooms-booking 6 register-book-for-meetings \[\] DINING-ROOMS \[\]

\( V \ d-r \ (\ d-r 6 DINING-ROOMS .^\text{card} (\ meeting-visits .^\text{-' } [\ dining-rooms-booking .^\text{-' } [\ \{\ d-r \}\ ]\ )\ )\ <\ Dining-Room-Capacity\ )\)

I N I T I A L I S A T I O N

conference-rooms-booking := {} II

dining-rooms-booking := {} II

O P E R A T I O N S

Create-Meeting

Create-Meeting ( meeting )

PRE meeting \in MEETINGS \[\]

meeting \^ register-book-\text{for-meetings} \[\]

THEN register-book-\text{for-meetings} := register-book-\text{for-meetings} U \[\] ( meeting \) END ;

Cancel-Meeting

Cancel-Meeting ( meeting )

PRE meeting \in register-\text{book-free-meetings} \[\]

meeting \{I. ran ( meeting-visits ) \[\]
meeting ? ran ( conference-rooms-booking ) ∧
meeting ^ dom ( dining-rooms-booking )
THEN register-book-for-meetings = register-book-for-meetings — { meeting } END ;

Add-Visitor-To-Meeting
Add-Visitor-To-Meeting ( visitor , meeting ) = PRE
meeting 6 register-book-for-meetings ∧ visitor £ register-
book-for-visitor ∧ visitor ^ dom ( meeting-visits ) ∧

Before we add a new visitor to the current meeting we want to be sure that there is as
minimum one vacant place in the dining rooms that were booked for this meeting.

card ( meeting-visits ^-1 [ dining-rooms-booking ^-1 [ dining-rooms-booking [ { meeting } ]]])< Dining-Room-Capacity
THEN meeting-visits := meeting-visits U { visitor i-> meeting }
END ;

Remove-Visitor-From-Meeting
Remove-Visitor-From-Meeting ( visitor ) PRE
visitor £ dom ( meeting-visits ) THEN

The result of this operation will be anti-restriction of meeting-visits by visitor (also known as domain
subtraction).
meeting-visits := { visitor } ^-1 meeting-visits END ;

Book-Conference-Room
Book-Conference-Room ( meeting , room ) =
PRE meeting 6 register-book-for-meetings ∧
room £ CONFERENCE-ROOMS ∧
room ^ dom ( conference-rooms-booking )
THEN conference-rooms-booking := conference-rooms-booking U ( room ^-1 meeting ) / END ;

Cancel_Conference_Room
Cancel_Conference_Room ( meeting ) ^
PRE meeting 6 ran ( conference-rooms-booking ) THEN

The result of this operation will be anti-co-restriction of meeting by conference-rooms-booking (also known as
range subtraction).

conference-rooms-booking := conference-rooms-booking ^ { meeting } END ;

Book_Dining_Room
Book_Dining_Room ( meeting , room ) =
PRE meeting £ register-book-for-meetings \ meeting ^
dom ( dining-rooms-booking ) ∧ room £ DINING-
ROO M S

Before we book the dining room we to want be sure that there are enough vacant seats for all visitors of the current meeting:

\[
\text{card } \{ \text{meeting-visits} \} \{ \text{dining-rooms-booking} \} \{ \text{room} \} \cup \{ \text{meeting} \} \}^\wedge
\]

**Dining-Room-Capacity**

**THEN** dining-rooms-booking := dining-rooms-booking \cup \{ \text{meeting} \} \text{=} \{ \text{room} \} \text{END} ;

**Cancel_Dining_Room**

**Cancel-Dining-Room** (meeting) =

**PRE** meeting \& dom (dining-rooms-booking)

**THEN** dining-rooms-booking := \{ meeting \} \setminus \text{dining-rooms-booking}

**END** ;

**Cancel-Meeting-Arrangements**

**Cancel-Meeting-Arrangements** (meeting) =

**PRE** meeting \& register-book-for-meetings

**THEN** register-book-for-meetings := register-book-for-meetings \&\& \{ meeting \} \setminus

conference-rooms-booking := conference-rooms-booking \setminus \{ meeting \} \setminus

\{ meeting \} \setminus \text{dining-rooms-booking} := \{ meeting \} \setminus \text{dining-rooms-booking}

**END** ;

**Create-Visitor**

**Create-Visitor** (person)

**PRE** person \& PEOPLE

**THEN** register-book-for-visiters := register-book-for-visiters \cup \{ person \} \text{END} ;

**Destroy-Visitor**

**Destroy-Visitor** (person)

**PRE** person \& register-book-for-visiters

**THEN** register-book-for-visiters := register-book-for-visiters \&\& \{ person \} \text{END} ;

**Book-Hotel-Room**

**Book-Hotel-Room** (person, room) =

**PRE** person \& register-book-for-visiters

**THEN** hotel-rooms-booking := hotel-rooms-booking \cup \{ person \} \text{END} ;
Cancel-Hotel-Room(person) =
  PRE  person 6 dom (hotel-rooms-booking)
  THEN  hotel-rooms-booking := \{person\} ~^\wedge hotel-rooms-booking
END

DEFINITIONS

HOTEL-ROOMS = N;
PEOPLE = N;
MEETINGS = N;
CONFERENCE-ROOMS = N;
DINING-ROOMS = N
END

5 Conclusions and Future Work

This paper is a preliminary step. Our main goal is object-based solution of ACVIS (A Computerized Visitor Information System). Here we achieved some results, which we can compare with object-based solution in purpose to find advantages and drawbacks later.

REFERENCES

8. The International B Conference Steering Comittee Web Site: http://www.sciences.univ-nantes.fr/asso/APCB/
10. The Z User Group Web Site: http://www.comlab.ox.ac.uk/archive/z/zug.html