ERA : KNOWLEDGE BASE FOR EPHEMERIS AND DYNAMICAL ASTRONOMY

G.A. KRASINSKY AND M.V. VASILYEV
Institute of Applied Astronomy, Russian Academy of Sciences
St. Petersburg, Russia

Abstract. A brief description of a problem-oriented language SLON for ephemeris and dynamical astronomy and a corresponding programming system ERA is given. In the frame of the system the user has easy access to an embedded universal program package to process high precision positional observations of Solar system bodies as well as of artificial Earth satellites.

1. Introduction

Now that new techniques of high precision astronomical observations generate an immense volume of observational data, a number of applied program packages are developed to process observations of different kind. For instance, there are program packages for geodynamical applications of Earth satellites and quasars, for investigations of spacecraft motion in deep space or in the vicinity of the Earth and planets, packages for lunar and planetary dynamics, for applications of pulsar timing and so on. As one can easily see, all the packages consist basically of similar program procedures and routines and only a small fraction of them are specific for each package. For example, any package is implemented by procedures to calculate precession and nutation or to transform the celestial reference frame to the terrestrial one and vice versa, by procedures to take into account the Earth orientation parameters etc. In fact, each of the packages could solve a considerably broader set of astronomical tasks than it has been designed for. However, in practice it is not easy to make an applied program package to deal with a task for which it has not been destined from the beginning. It seems clear that it is possible to construct a universal program package for any type of ephemeris applications; the real problem is not to *compile* such

I. M. Wytrzyszczak, J. H. Lieske and R. A. Feldman (eds.),
Dynamics and Astrometry of Natural and Artificial Celestial Bodies, 239, 1997.
© 1997 *Kluwer Academic Publishers. Printed in the Netherlands.*

a package but to invent an easy way in which the user could *work* with it. In Krasinsky *et al.* (1989) an approach based on using a special problem-oriented language SLON designed for ephemeris and dynamical astronomy was proposed, and a corresponding programming system ERA [Ephemeris Research in Astronomy] was outlined. SLON is the successor to a language called MAMONT, which is an acronym for *MA*thematical *MO*deling for Celestial Bodies (*Nebesnikh Tel*) in Russian. The current version ERA-6 of the system is essentially more advanced both in the functional power of the applied program package and in the descriptive strength of the language SLON. The system has been tested by a number of applications and has proved efficiency of the approach.

The main feature of the system is its versatility: the user is not limited by a predefined set of ephemeris tasks but can easily develop his own applications practically in any branch of positional astronomy.

2. Problem-Oriented Language SLON for Ephemeris Astronomy

It is supposed that the main aim of the user is to prepare ephemeris predictions for an astronomical observational campaign and (if the product of the campaign is accurate position observations) to process data obtained. When programming in SLON language the user has to describe the observational program under consideration in terms of more or less adopted astronomical notation; after that, all ephemeris calculations will be produced automatically. For instance, the observed astronomical object must be specified as a variable **object** (its value must be chosen from a predefined set of literals **Sun**, **Moon** etc); the type of the observations as a variable **observation** with values from another set of literals (**spherical**, **ranging** etc); the coordinate system as a variable **coordinates** (values **equatorial**, **horizontal** etc) and so on. If some of the specifications are omitted then the missing variables will assume their default values. As a rule in SLON programs, the majority of the variables employ the default values and the descriptions appear to have a very compact form.

Due to default conventions a meaningful astronomical observation will be always defined when arbitrary values are assigned to any subset of the SLON variables (if the values are matched to types of the variables). Thus, an "elementary" ephemeris task arises and corresponding ephemeris predictions may be calculated if the user applies a procedure **compute** (without any arguments) which automatically calls procedures which are needed from the applied program package to solve the astronomical task defined in this way.

The most specific feature of the language SLON is a way in which it generates a flow of the "elementary" tasks; such a flow may correspond to a rather sophisticated observational program in which a number of observers or/and a number of observed bodies are involved.

The simplest way to generate such a flow is to define a table which contains values of the variables for each elementary task in every line ("tuple" in terms of relational databases). The table may be modified and saved by built-in operations of relational algebra; in this way the SLON user may create and manage his own databases, for instance to keep and retrieve astronomical observations.

To describe any task of ephemeris astronomy it is necessary to make use of a set of variables of different types: they may be integers (or long integers), reals (as well as reals of double and extended precision), booleans, strings (or long strings); they also may be variables of literal (numerable) types (their values belonging to a finite set of literals). As a high-level language, SLON strictly controls types of variables. The variables may enter arithmetical or boolean expressions and be arguments of built-in functions and procedures.

As a simple example let us consider a SLON program that calculates apparent positions of the Sun, Moon and Venus for some calendar date:

```
EXAMPLE:=//object, cx, cy,  calend(date)/
            Sun   , * , * ,  19940101.12/
           Moon   , * , * ,  19940101.12/
           Venus  , * , * ,  19940101.12//
           begin
            compute
            writeln( object, cx:hms, cy:dms, date)
           end.
```

The program defines a table EXAMPLE (a "relation" in terms of relational databases) with a header and three lines ("tuples"). The name of the variable for each field is defined by the corresponding literal at the header; values of the variables are given at the tuples. In this example the variables are **object** (one of numerable literal types), **cx**, **cy** (double precision) and **date** (extended precision). The first column (field) of the table corresponds to the variable **object**; its value is **Sun** at the first tuple, **Moon** at the second, and **Venus** at the last one. The second and third columns contain variables **cx** and **cy** which have the same value $*$ ("asterisk") in each tuple. That is an indication for the procedure **compute** to place the calculated right ascension instead of **cx** and the declination instead of **cy**.

Values of the variable **date** enter the fourth column in a presentation "calendar date" [i.e. as a packed number in which the integer part consists of the year (1994), the month number (01), the day number (01) and the fractional part consisting of the hours (12), minutes (00) and seconds (00)].

The header and tuples are separated by the symbol "/"; the symbol "//" opens the header and closes the last tuple.

The sequence of procedures which are limited by operational braces **begin** and **end**. In the example they are **compute** and **writeln**, which are called at each of the tuples of the table EXAMPLE. Thus, after the program has been run the resulting table EXAMPLE contains calculated apparent equatorial coordinates of the objects (as **cx** and **cy**); while the program is running the Pascal-like procedure **writeln** outputs values of its argument on the monitor in the indicated format.

Because the variables **date**, **cx** and **cy** have the same values at each tuple, the table EXAMPLE may be "factorized" and an equivalent SLON program might be written in more compact form:

```
EXAMPLE:=
 (//cx, cy,  calend(date)/
    *,   *, 19940101.12//
    *
  //object/   Sun/ Moon / Venus //)
    begin
     compute
     writeln( object, cx:hms, cy:dms, date)
   end.
```

That is a very particular example of a relational algebra used by SLON to handle tables. Here, the symbol "asterisk" $*$ between two tables stands for an operation of "multiplication" of two tables; other useful operations are "addition" of tables and "iteration" of tables (multiplication of tables by integer numbers).

3. Types of Observations

It appears that practically all variety of types of positional observations may be described in rather simple and concise way if one chooses a correspon-ding value of the variable **observation** from the following set: **spherical** (spherical longitude and latitude in the coordinate system defined by the variable **coordinates**); **differential** (differences of longitudes and latitudes of two observed objects on the celestial sphere); **angular_distance** (angular distance and position angles of these bodies); **phase_angle** (angle between directions from the first observed body and the second one); **occultation** (angular distance between apparent limbs of two bodies and time-derivative of this value); **eclipse** (angular distance between apparent limb of the first body and a boundary of the shadow of the second body); **apparent_radius** (apparent radius of the observed body); **ranging** (two-way range to the observed body and its rate); **pseudorange** (one-way range and its rate); **interferometry** (interferometric time delay and its rate).

In total there are more than 50 variables needed to describe in full all the variety of positional observations; as a rule it is enough to define explicitly only 5-7 of them (due to the default conventions).

After defining an astronomical task in this way the procedure **compute** may calculate ephemeris values of corresponding observables. When possible, the calculations are made in accordance with IERS standards (1992).

4. Dynamical Models in ERA-6

In the examples of the SLON programs given above there are no references to dynamical models. It means that the procedure **compute** will make use of built-in dynamical models for the observed bodies (as well as for the observer). The models may be given either as numerical theories in the form of Chebyshev polynomial representation (that is the case of lunar and planetary ephemerides) or in the form of analytical theories (for main satellites of major planets). Embedded theories are the following ones: satellites of Mars: Phobos, Deimos (Chapront-Touzé, 1988); Galilean satellites of Jupiter: Io, Europa, Ganymede, Callisto (Lieske, 1977); nearest satellites of Jupiter: Amalthea, Himalia (Rohde and Sinclair, 1992); faint satellites of Jupiter: Elara, Pasiphae, Sinope, Lysithea, Carme, Ananke, Leda (numerical model); satellites of Saturn: Mimas, Enceladus, Tethys, Dione, Rhea Titan, Iapetus (Vienne and Duriez, 1995), Hyperion (Dourneau, 1987); satellites of Uranus: Ariel, Umbriel, Titania, Oberon, Miranda (Laskar and Jacobson, 1987); satellites of Neptune: Triton, Nereid (Rohde and Sinclair, 1992); satellite of Pluto: Charon (Harrington and Christy, 1981).

The user can construct his own dynamical theory with the help of a unified procedure of numerical integration **integrate**. Like the procedure **compute** it has no explicit arguments and takes information needed to work from a set of variables which describe the process of numerical integration. The variables indicate the object (or objects) for which the dynamical theory must be constructed, initial values, precision, time span and a number of other parameters (values of most of them may be taken by default conventions). For instance, the type of equations is defined by the variable **equations** with the following values: **single_body** (equations of motions of a single body), **solar_system** (major planets, Moon, rotations of the Moon and Earth), **Jovian_satellites** (the system of satellites of Jupiter) and analogous values for other systems of satellites. Results of the integrations are saved in the form of Chebyshev polynomials. The same procedure also integrates variational equations for a number of parameters.

For the value **single_body** the body for which the equations of motion will be integrated may be a planet, comet, spacecraft on heliocentric orbit, Earth satellite, satellite of other planets and Moon. In the case of Earth satellites the dynamical model is accurate enough for geodynamical applications (SLR or GPS techniques).

If the value is **solar_system** then the equations of motion of the Moon and planets will be integrated by the method used in the ephemerides DE200 (IERS 1992) (with some developments).

5. Conclusion

Our experience has proved that the programming system based on the proposed approach gives a useful tool for scientific work in ephemeris and dynamical astronomy. Because a scope of its applications is practically unlimited the testing is a continuous work which never may be considered as completed. At this colloquium two papers have been presented which are examples of such a testing work with a scientific issue: Aleshkina *et al.* (1996) on LLR observations, and Pitjeva (1996) on ranging to Martian landers. The software may be obtained from the authors by request.

References

Aleshkina, E.Yu., Krasinsky, G.A., and Vasilyev, M.V.: 1996, "Analysis of LLR data by the program system ERA", this volume.

Chapront-Touzé, M.: 1988, personal communication.

Dourneau, G.: 1987, "Observations et étude du movement des huit premiers satellites de Saturn", Ph.D. thesis, Université de Bordeaux.

Harrington, R. and Christy, J.W.: 1981, "The Satellite of Pluto", *Astron. J.* **86**, 443.

Krasinsky, G.A., Novikov, F.A., and Skripnichenko, V.I.: 1989, "Problem oriented language for ephemeris astronomy and its realization in the system ERA", *Celest. Mech.* **45**, 219–229.

Laskar, J. and Jacobson, R.A.: 1987, "GUST86. An analytical ephemeris of the Uranian satellites", *Astron. Astrophys.* **247**, 565.

Lieske, J.H.: 1977, "Theory of motion of Jupiter's Galilean satellites", *Astron. Astrophys.* **56**, 333–352.

McCarthy, D.D., (ed.): 1992, "IERS Standards (1992)", *IERS Tech. Note 13*, Obs. de Paris.

Newhall, X X, Standish, E.M., and Williams, J.G.: 1983, "DE102: a numerically integrated ephemeris of the Moon and planets spanning forty-four centuries", *Astron. Astrophys.* **125**, 150–167.

Pitjeva, E.V.: 1996, "The ephemerides of the inner planets from spacecraft range data and radar observations 1961–1965", this volume.

Rohde, J.R. and Sinclair, A.T.: 1992, "Orbital ephemerides and rings of satellites", in: *Explanatory Supplement to the Astronomical Almanac* (P.K. Seidelmann, ed.), University Science Books, Mill Valley, California.

Vienne, A. and Duriez, L.: 1995, "TASS 1.6: Ephemerides of the major Saturnian satellites", *Astron. Astrophys.* **297**, 588–605.