Supporting Complex Flight Scheduling Tasks Using CLOS and Statice

Matthias Grund, Gerd Timmermann*

Symbolics Systemhaus GmbH, Germany
Mergenthalerallee 77-81
W-6236 Eschborn / Ts.

Keywords: object-oriented concepts, protoyping, downsizing, oodb

1. Introduction

In the context of the actual trends towards downsizing and client-server architectures, new opportunities and also new requirements to integrate innovative solutions into conventional environments arise. In the future workstations will not just handle office activities and office communications, but moreover services like decision support systems and complex information systems will be available in a transparent way.

The client-server applications presented here are planning support systems with a graphical, interactive user interface. Powerful tools had to be created that do not confine the users in their autonomy, but ensure both an acceleration of planning sequences and the consistency of atomic planning tasks.

Hardware base of the systems are SUN4 workstations with Symbolics coprocessors, used as servers for the Lisp-based object-oriented database Statice. The intelligent graphical user interfaces run on SPARC-stations and Macintosh-PCs. The applications have been written consistently in Lisp, from the database to the user interface. The database server communicates with several mainframes.The system configuration is given in figure 1.

2. Protoyping and Object-Orientation

2.1 Utilizing the Competence of Users in the Software Development Process

Downsizing actually does not just mean to transfer existing solutions to new hardware platforms, but moreover to adjust the application development process to business organizations and to business administration sequences[1]. New computer systems will have to support these tasks in an optimal way. This does not only require to incorporate the users into the development process, those who are operating in these sequences and do

* The authors can be reached via phone: x49/6196/47220, fax: x49/6196/481116,

e-mail: mg%sger.uucp@Germany.Eu.net, gt%sger.uucp@Germany.Eu.net

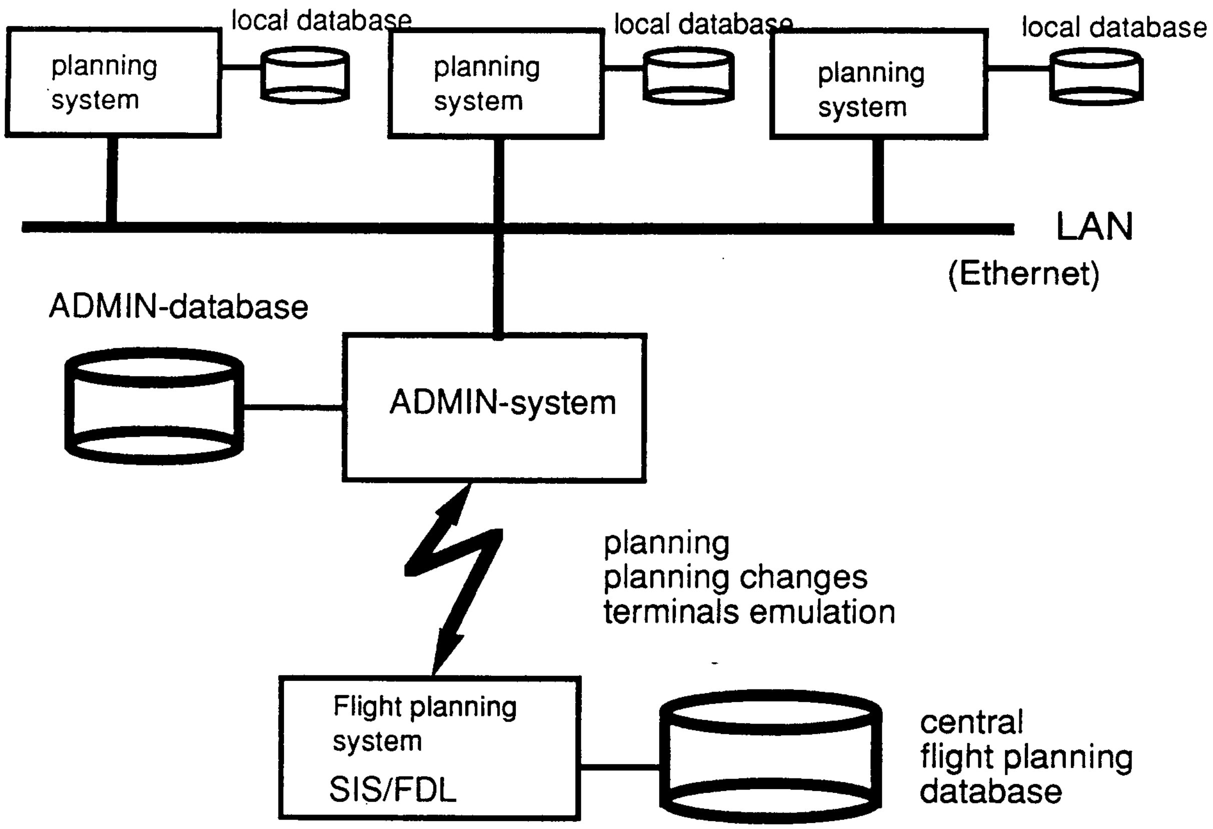


fig. 1 System configuration

know and control the formal and especially the informal structures. It also demands to integrate the specialists into the conception, implementation and further development of those systems as directing and controlling authorities.

The necessity of integrating the users more than in the past corresponds to an increased user competence in data processing subjects. Competence does not only include the practice to use computers but also the increased ability to think in terms of edp. Both events are a result of a decade of personal computing and an increased significance of computer science in education. Most graduates have learned computer science at least as a subsidiary subject these days.

2.2 New Techniques Offer New Opportunities

In parallel to the diffusion of edp knowledge, the modelling techniques used in software development have been approximated to human ways of thinking. Entity relationship models and especially object-oriented models can be discussed with a competent user. For this reason it has become possible that the users assist in the critical phases of analysis and design.

This early participation not only is possible, but is asked for by emancipated users and is pushed for the following reason: Many analyses of failed or stagnating large projects have shown that in the course of software production the most serious errors have not been of technical nature but have occurred in the early phases of analysis and design. These errors are hardly ever caused by deficiencies in the technical skills of the participants but by insufficient communication between users and developers.

In the context of the new technological possibilities and the new economical requirements, prototyping as a method of software development achieves a new dimensionality.

2.3 Prototyping Revisited

Former prototyping concepts concentrated in building user interfaces, in exploring possibilities and risks of concepts to realize a project, and as well in generally proving feasibility. Nowadays the support of the critical phases of building a model is becoming more and more important. Building a model is the task to be accomplished by the classic phases of analysis and design. Evolutionary aspects of prototyping are clearly becoming more significant than explorative procedures. With the ambition to exhaust the possibilities of object-oriented concepts by modified software development schemes, Symbolics Systemhaus GmbH developed the method of regulated prototyping[2].

Without going too much into details it should be mentioned, that the heart of this method is to build a model of the problem field and of the solution jointly by both users and developers. Just like the model of a building, in this method the prototype serves as a catalyst for the communication between users and developers as well as a clearing instance between the participating user departments. The communication is based upon an early system with a user interface that covers most of the essential parts completed by a graphical notation of the object model. The intuitive accessibility of the graphical notation for an object-oriented model enables this form of participation in early but crucial project phases. An early implementation of the user interface not only serves the goal to achieve an ergonomic system but also to support the development of the underlying object model.

3. Examples of Real World Projects

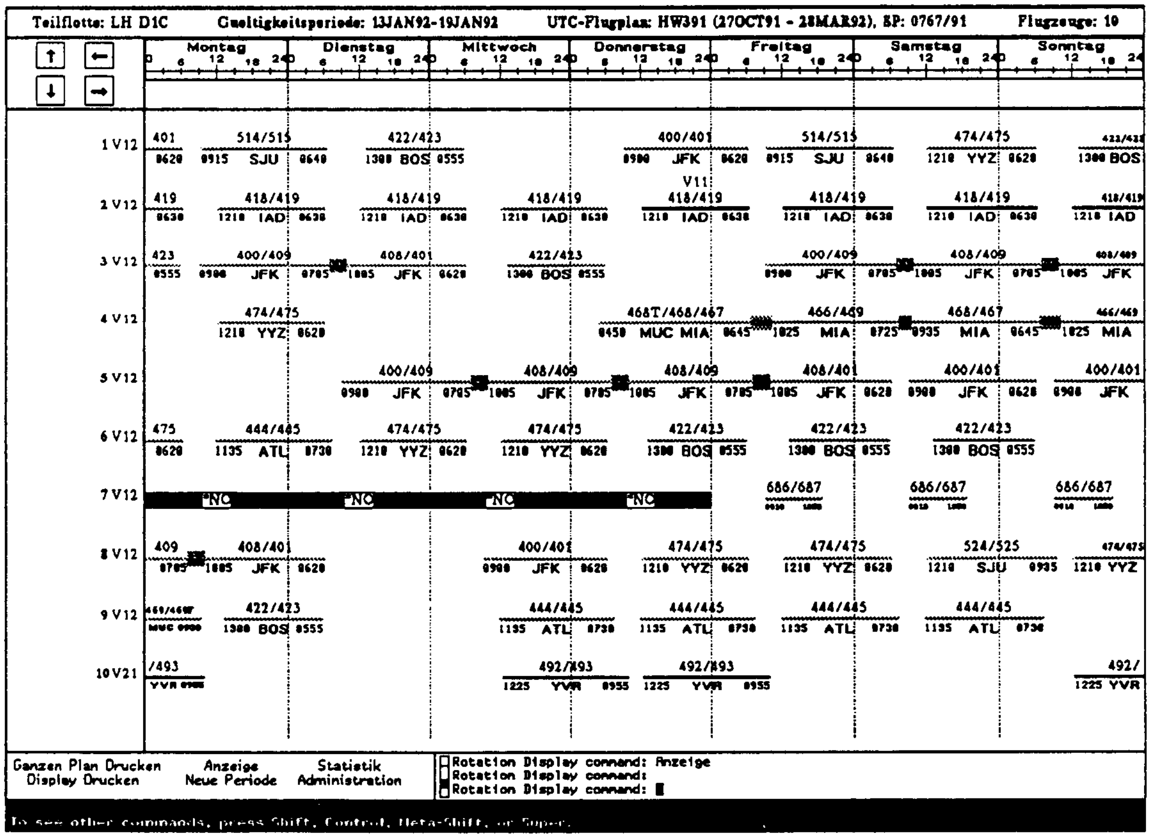
The Symbolics Systemhaus GmbH specializes upon solutions in the field of planning and disposition, diagnosis and configuration. Our experiences in employing object-oriented techniques in a cooperative development environment will be conveyed by giving two examples for solutions in the field of production planning of a big German airline (Deutsche Lufthansa AG). The production planning process at Deutsche Lufthansa AG can be divided roughly into the following three steps:

- product planning to determine an economical offering based upon demand forecasts and market assessments

- flight and capacity planning to determine the optimal but feasible offering and the fleet structure needed for its realization
- ressource planning (e.g. maintenance planning, crew employment planning, airport capacities, a.s.o.)

Production planning starts with a target schedule obtained from the product management that contains the desired offering of flights together with their routings, operating frequencies, approximated times, and aircraft capacities.

Originating from that target schedule, flight and capacity planning determines the needed aircrafts (number and size), the optimal fleet structure (capacity planning), and converts the target schedule together with the fleet structure into a most optimal production schedule, always considering internal and external restrictions (flight planning). The production schedule includes a rotation schedule (fig. 2) that specifies, which flight should be serviced by which "fictive" aircraft, and a flight schedule, that contains all flights with accurate descriptions (operating frequencies, times, aircraft types, a.s.o.). The flight schedule is the mandatory foundation for production and sales as well as for all company activities based thereon.


Teilflotte: LH D1C Gueltigkeitsperiode: 13JAN92-19JAN92 UTC-Flugplan: HW391 (27OCT91 - 28MAR92), SP: 0767/91 Flugzeuge: 10

Ganzen Plan Drucken Anzeige Statistik
Display Drucken Neue Periode Administration

Rotation Display command: Anzeige
Rotation Display command:
Rotation Display command:

To see other commands, press Shift, Control, Meta-Shift, or Super.

fig. 2 Part of rotation schedule

Ressource planning takes the rotation and flight schedules and developes production plans for all departments participating in realizing the flight schedule like maintenance planning, aircraft usage planning, crew employment planning, airport capacity planning, material planning, a.s.o.

The schedules are made up of around 220 aircrafts serving around 200 airports worldwide. The rotations have to be varied on a weekly, sometimes daily basis to achieve an optimal adjustment to seasonal actualities or national and international events, e.g. fairs.

The complexity of the planning tasks is caused by a relatively long planning period (several years), by a continuous adjustment of the schedules to market demands, by different planning restrictions in different departments (e.g. constrained capacities due to maintenance for other airlines), and by various feedbacks between all steps of the planning process (fig. 3).

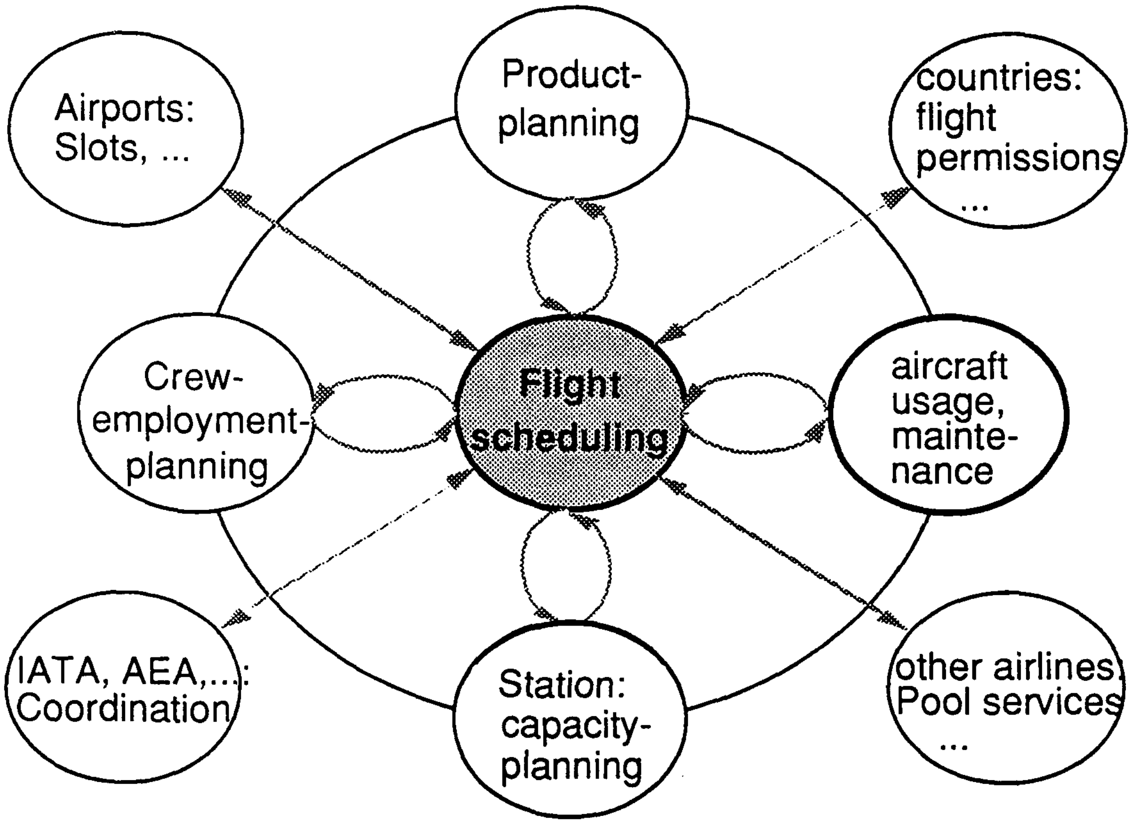

Airports: Slots, ...
Product-planning
countries: flight permissions ...
Crew-employment-planning
Flight scheduling
aircraft usage, mainte-nance
IATA, AEA,...: Coordination
Station: capacity-planning
other airlines: Pool services ...

fig. 3 Influences on flight scheduling

Today, production planning is still done mainly without computer support. All data are stored in mainframe databases, semi-graphical listings are processed manually (fig. 4), the results are entered manually on ASCII-terminals, with the immanent danger of inconsistencies and oversights. Some departments are developing or introducing planning support systems in order to be able to manage the planning process, which is getting more and more complex, due to growing aircraft fleets and tendencies towards more flexible flight schedules. Two central departments (flight and capacity planning[3] as well as maintenance planning[4]) are using the object-oriented database system Statice for the systems currently being developed (parts of the systems are in operation right now).

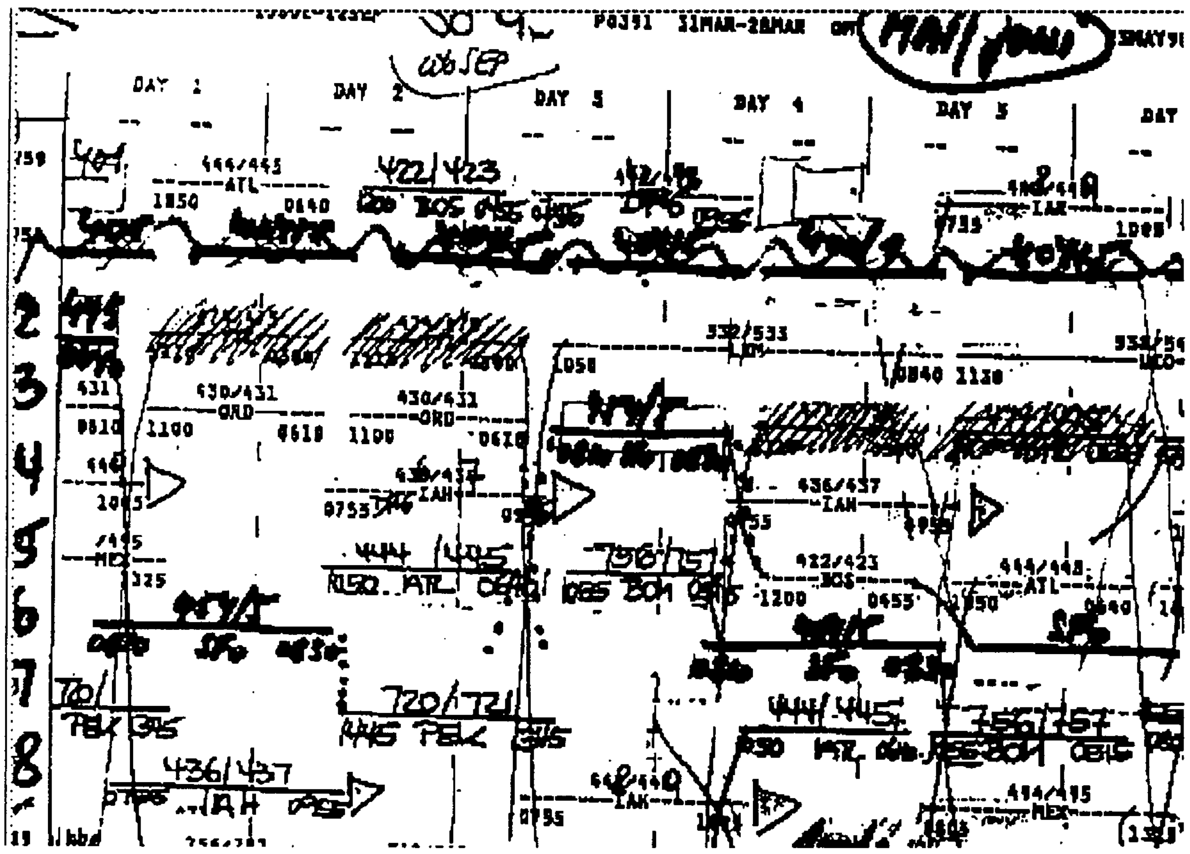


fig. 4 **Manipulation of s.g. computer listings as done in the past**

Deutsche Lufthansa AG decided to use the object-oriented database Statice after a comparative evaluation including adequate performance tests. The results showed that a relational database system could not comply with the performance requests concerning the data processing needed for interactive changes of the planning data.

From a technical point of view, in addition to performance, the modeling opportunities given by an object-oriented concept have been of major interest. An excerpt of the data model for a flight schedule is shown in fig. 5.

From the users' point of view, besides performance, the possibility to perform complex and consistent modifying operations via an interactive graphical user interface (fig. 6), and the availability of several graphical, semi-graphical, and tabular outputs needed for the different departments envolved in the planning process, have been significant.

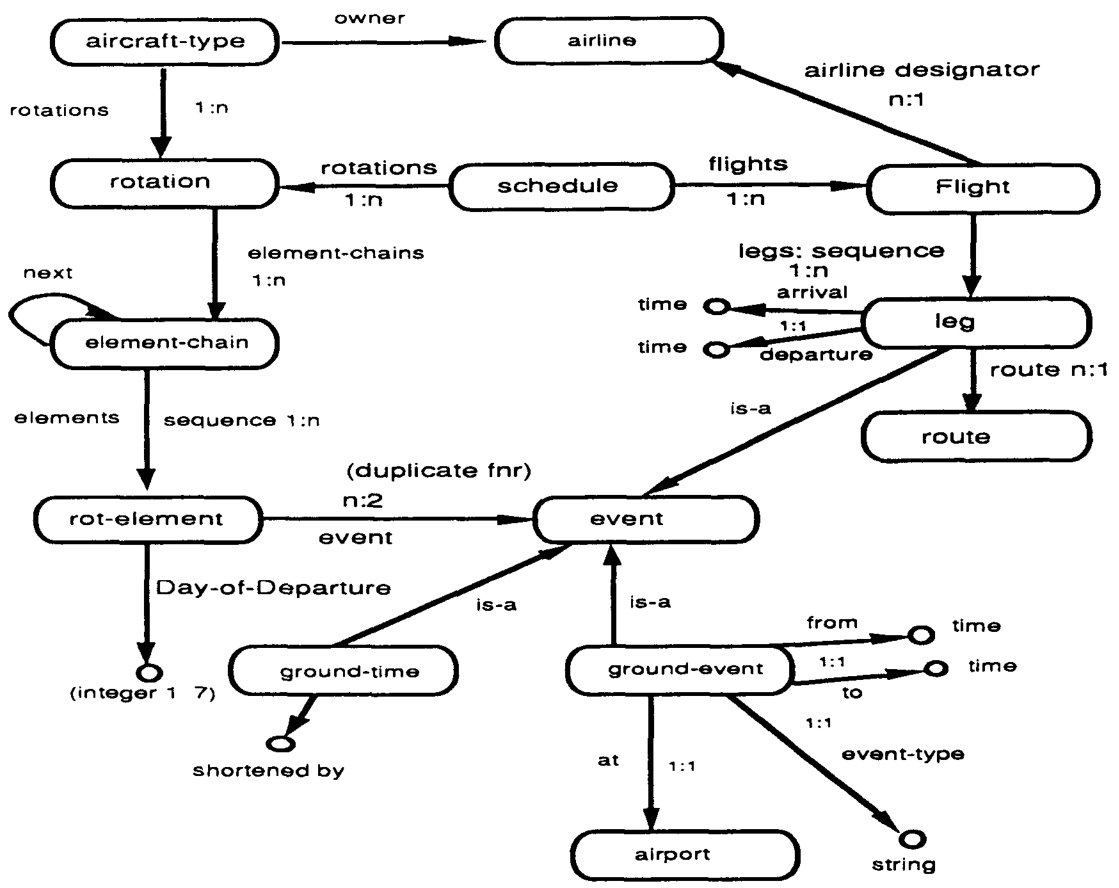


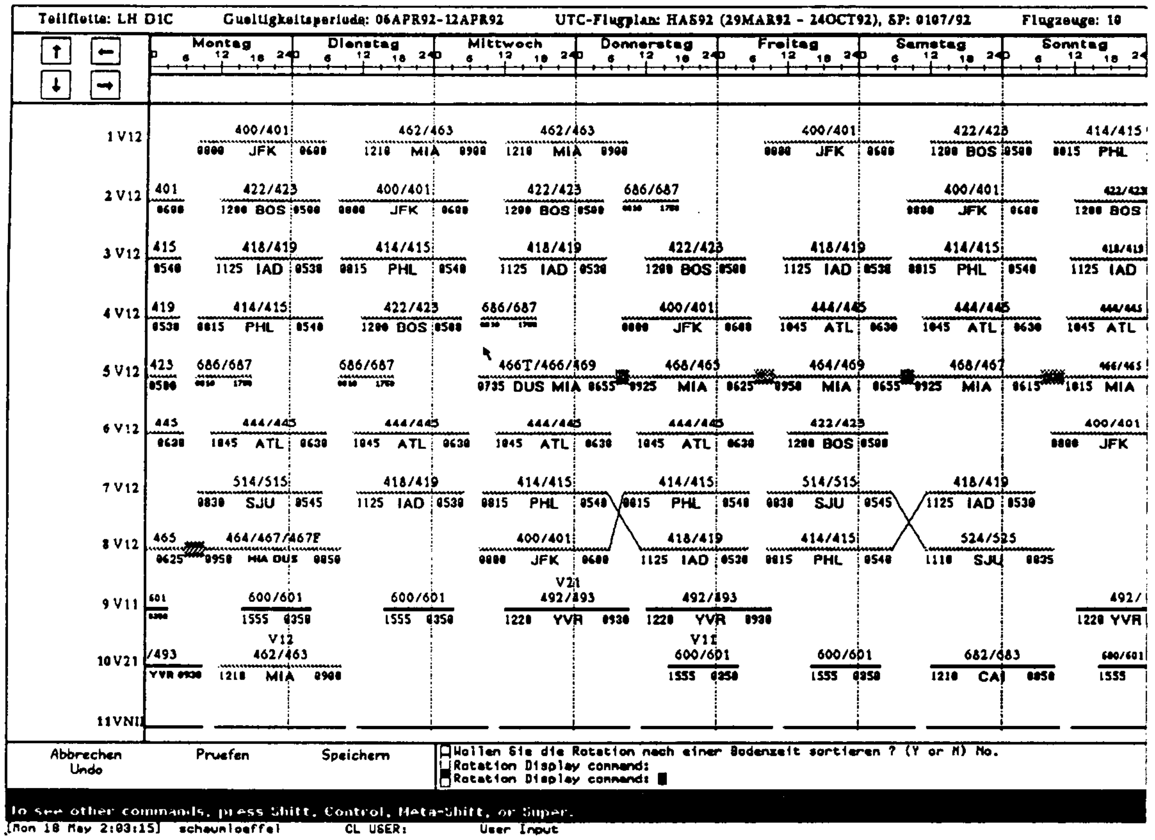
fig. 5 Excerpt from the data model for flight plans

Working on a flight schedule is a lengthy task, often performed by several experts in parallel. So, in addition to the obvious multi-user capabilities, support for long transactions and for the management of alternatives and versions of the data are requirements to the system.

Since the Deutsche Lufthansa's central data storage is done on mainframes, another demand to the system was its integration into the existing computer environment.

In the beginning both the graphical interface and the set of modifying operations have been oriented on the traditional planning documents available in paper form and on the practiced series of operations. While the system matured, new wishes arose, together with new ideas about new possibilities that should be offered by the system.

The total volume of one of the projects described above has been about 6 million dollars.



```
Teilflotte: LH D1C   Gueltigkeitsperiode: 06APR92-12APR92   UTC-Flugplan: HAS92 (29MAR92 - 24OCT92), SP: 0107/92   Flugzeuge: 10
```

```
Wollen Sie die Rotation nach einer Bodenzeit sortieren ? (Y or N) No.
Rotation Display command:
Rotation Display command:
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Mon 18 May 2:03:15]   schaunloeffel        CL USER:        User Input
```

fig. 6   The crossed lines show the result of a temporary change in 2 weeks

4. Lisp - The Language for Evolutionary Software Development

The decomposing, classifying, and structuring methods used in object-oriented design are guided by the perception of reality by human intelligence and thus are easily accessible to intuition. Therefore - as shown before - the application of object-oriented techniques offer the possibility to thoroughly integrate the users into problem analysis and modelling.

The consistency of structural elements (object, class, method, relation) from the object-oriented analysis through object-oriented design to object-oriented program code allows to dissolve and shift the stiff boundaries between each phase. Furthermore, the consistency prevents semantic breaks that occur, if a transition between phases involves transformation between structural concepts, e.g. from entity relationship model to relational schemata in first normal form.

In addition to this horizontal consistency of object-oriented concepts from analysis to coding, the vertical consistency of object structures from the database through application code to the user interface is important[5]. Exactly for this pupose, Lisp offers by its presentation type concept of CLIM an industry standard not found in any other environment.

By developing standards like Common Lisp, CLOS, CLIM, by developing a sophisticated condition handling system, Lisp has envolved to a programming language well suited for use in industrial software production.

Furthermore, Lisp is an immense mighty and expressive multi-purpose language. It has been developed as a symbol processing language. Symbol processing together with the paradigm of object-orientedness guarantees program code with a high congruence to the modelled reality. Reality is sufficiently complex. System development should not be burdened by additional complexities effected by inadequate languages.

In the context of complex applications that need an evolutionary system development and the possibility of quick exploration of solution ideas, Lisp seems to be downright predestinated. The modularity encouraged by Lisp allows rapid prototyping with simplified implementations in varying levels of concreteness[6]. A step-by-step refinement of code is easy and is supported by incremental compiling. Sophisticated development environments, being tradition in the Lisp world (Interlisp D of Xerox Parc or Genera of Symbolics) allow inexpensive prototyping.

By its functional nature, Lisp promotes code reusability through a less imperative style of programming[6]. Lisp is equipped with a mature and standardized object system, CLOS[7](Common Lisp Object System), that originates in ancestors like flavors[8] (MIT) and LOOPS[9](Xerox Parc) and therefore is built up on more than a decade of experiences. In the applications discussed herein, the object-oriented database Statice has been employed, which is well integrated into the Lisp object modell and the Genera environment.

For now, project experiences prove the correctness of a cooperative and evolutionary system design. The system is highly approved. The level of standardization achieved and the tools available in the Lisp environment make Lisp a useful language for developing applications, that cannot be delivered using conventional standard tools and methods. The solution of complex problems, that have a demand for evolutionary system development involving the user, remains a domain of Lisp.

Lisps's image as a language usable for real life programs suffered in the last decade from to the disappointments that occurred in the field of artificial intelligence. However, with a propagation of the paradigm of object-oriented concepts, analysts[10] expect that Common Lisp CLOS together with CLOS-based object-oriented databases will not only gain ground again, but moreover will be able to soundly take a big market segment. This will particularly be the case, if tools for the Lisp market get developed, that support the capabilities of Lisp in the areaof evolutionary system development even better than today.

According to our opinion, the next logical steps should support the fast development of graphical user interfaces and object-oriented models.

To support the process of prototyping, we first need a tool to quickly generate user interfaces in an interactive graphic way. The protoyping of user interfaces certainly is not the focus of interest in controlled prototyping, but still is an inevitably essential part of it. Second, a tool to graphically develop object models with automatic code generation (database schemata and class definitions) is needed. Both tools should be integrated into existing environments. A next step could be the management of a repository based on a object-oriented database. Beyond that, it would be nice to have a documentation system integrated into this environment. Symbolics Systemhaus is already working on some of these areas, internally or with partners.

The positioning of Lisp as a high end programming language for high end jobs is evident. Focusing development efforts upon tools mentioned above, will assist the establishment in this market segment.

References

[1] Jacobi A.: Objektorientierung und Downsizing; komplementäre Paradigmen, Hansen (ed.) Downsizing, Grundlagen, Realisierungswege, Erfolgsfaktoren, München, 1992

[2] Grund M., Popp C.: Reguliertes Prototyping, internal paper of Symbolics Systemhaus GmbH, publication in preparation

[3] FrankenR.: Designkonzept eines wissensbasierten Planungsunterstützungssystems, IST 4/91

[4] Stahl J., Grund M. Objektorientierte Datenbanken im Einsatz; IST 4/91

[5] Fischer Markus: Unifying Software Elements with Lisp-Based Object-Oriented Technology, LUV 92 (this volume)

[6] Sinclair K., Moon D. A. The Philosophy of Lisp, Common ACM 34, 9, Sept. 1992

[7] Keene Sonya E.: Object-Oriented Programming in Common Lisp - A Programmer's Guide to CLOS

[8] Cannon H. I. : Flavors: A Non-Hierarchical Approach to Object-Oriented Programming, 1982

[9] Bobrow D. G., Stefik M.: The LOOPS Manual, Xerox Palo Alto Research Center, 1983

[10] Jeffcoate J., Templeton A.: OVUM Report: Object Technology Sourcebook, 1991 OVUM Ltd.