Semi-automatical Forecast Production in Java

D. Matter, P. Ambrosetti, P. Müller, Ch. Pauli
Swiss Meteorological Institute
Krähbühlstr. 58
CH-8044 Zurich

1 Summary

The increase of input data for weather forecasing and the increase of the number of products demands a rationalisation of the production process. We describe here the situation at the Swiss Meteorological Institute with our approach to a solution. We introduce a forecast database, that will be filled once and used many times for most our products. The forecast editing programm has been written entirely in Java.

2 Weather Forecast vs Production

In the last years all meteorological services saw a dramatic increase of the amount of data, both of observations and numerical model outputs, and the number of forecast products. Today weather services are able to forecast more parameters with increased time and space resolution for longer periods. Our customers require specially tailored products as cheap as possible (because of the competition with private forecast services). The bottleneck is the forecaster: He gets more and more data and he must produce more and more products (Figure 1.a), increasing the forecast quality at the same time. This is not an easy challenge!

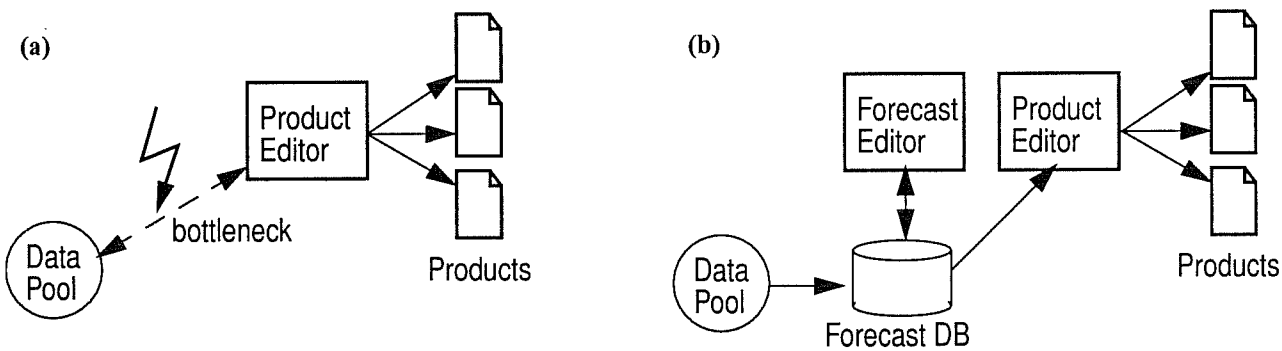


FIGURE 1. (a) Todays Production Process -- (b) New Production Process.

One first step towards a solution is to separate the forecasting step from the production step itself (Figure 1b). In the forecasting step the forecaster uses a Forecast Editor to produce and modify weather elements which are stored in a database (Forecast DB). These elements will then be used during the production step. Going this way the same forecast can be used for several products avoiding a lot of repetitive work and assuring consistency among different products. This concept frees forecasters from preparing routine products and allows them to concentrate on studying the large amount of input data. Also the forecaster will be able to better issue severe weather warnings. A monitoring application on the Forecast DB may then compare the incoming weather observations to

the current forecast. If they differ significantly an alert may be presented to the forecaster. Moreover the Forecast DB will support enhanced verification procedures.

Several weather services (both public and private) developed different approaches to fill Forecast DBs with data. The simplest one is "to switch off" the forecaster and to produce forecasts by MOS techniques directly from the models. This often results in poor forecasts particularly over complex terrain like the alpine region.
Another approach is to edit the model output fields and/or time series (like the Canadian Forecast Production Assistant or the U.S. AWIPS) and then feed the results into the DB. This approach is best for weather services with a centralized forecast production, because it is difficult to achieve consistency among fields edited by several weather centers. We think, however, that this approach would not work at all in Switzerland where one can experience very strong meteorological gradients e.g. between the north and south slopes of the Alps.

For these reasons we developed a "hybrid" approach, more suitable to our way of producing forecasts and using the large experience of our forecasters, who have detailled knowledge of the regional meteorological peculiarities.

3 The Swiss "philosophy"

3.1 Requirements

First we identified the requirements for the forecasting step in our weather offices:

- The whole step (connect to the database, edit, check, store the data) must be simple, quick and user friendly (no manual has to be consulted during the work).
- The forecaster must keep the control over all the forecasts.
- Consistency in time and space must be guaranteed, as well as among the different parameters.
- Very sharp meteorological gradients should be possible. Switzerland is a small country, but with a lot of "local weathers", where the altitude plays an important role.
- Forecasts must be verifiable against observational data from our dense meteorological networks.
- Since the late '70ies the SMI operates a dense network of automatical meteorological stations, well distributed all over the country. The forecasters acquired a detailed knowledge of the local weather effects, because they can follow the meteorological development with a high time resolution (10 minutes). This kind of knowledge should be embedded in this new forecast tool.
- All (or at least most) of the elements in the forecast database should be used in products.
- Only relevant parameters are in the database.
- The forecast accuracy both in space and time decreases with the range. (It is meaningless to fill a forecast DB with data that cannot be forecasted with a reasonable quality.)
- The user interface must be suitable for both editing and looking at the elements of the database. (In this way it will be possible to quickly access the data for personalized telephone forecasts.)
- The forecast editing should be as close as possible to the mental forecasting process, i.e. to the time evolution of weather systems (like fronts) over an area. Forecasters are used to look at time sequences of satellite/radar pictures, weather maps and computed fields from numerical models.

3.2 Content of the Forecast Database

We analyzed several possible solutions for the organization of the forecast data and we decided to go our own way. We are going to fill our forecast DB with data for predefined regions and time intervals for a selected number of parameters. The following selections have been made for the regions, time intervals and parameters:

3.2.1 Regions

- For the short range (0 - 48 hours) we selected 36 regions, where at least one automatical station is present (usually several at different altitudes in a region). Specific climatological knowledge has been used in the definition of these regions.
- For the near mid-range (48h - 96h) the previous regions have been grouped into 9 small clusters.
- For the far mid-range (96h - 216h) 7 larger clusters were defined.

3.2.2 Time intervals

- For the short range 6 hours intervals have been selected.
- Near mid-range: 12 hours.
- Far mid-range: 24 hours.

3.2.3 Weather parameters

- Temperature (low- and mid-altitude ground temperature or Min/Max)
- Cloudiness (in classes)
- Significant Weather (Snow, Thunderstorm, Freezing rain, etc.)
- Accumulated precipitation in the interval
- Probability of Precipitation
- Wind (Speed and Direction)
- Rain/snow limit
- Top of low cloud/fog
- Confidence Index
- Just for 3 clusters: Temperature and wind at 1000 m, 2000 m, 3000 m, 4000 m altitude

The forecast DB will be initialized with DMO/MOS data from different numerical models (Swiss Model for the first 48 hours, ECMWF later on). At least in the beginning we will not fill the database with nowcasting data for the whole country.

The forecast data are presented to the forecaster in a pseudo-synoptic form, i.e. for each region/station a set of numbers and symbols will be displayed. The forecast editing program (see below) is written in such a way, that it is easy to change the regions, range and parameters if our experience, knowledge and necessities change.

4 The Swiss Forecast Editor: GUI and functionality

The Forecast Editor (FE) is fully written in Java. Its main window consists of four main parts: a geographical display of the regions, a time interval step, a parameter and a value selection. The following sections describe the functionality of each of these parts.

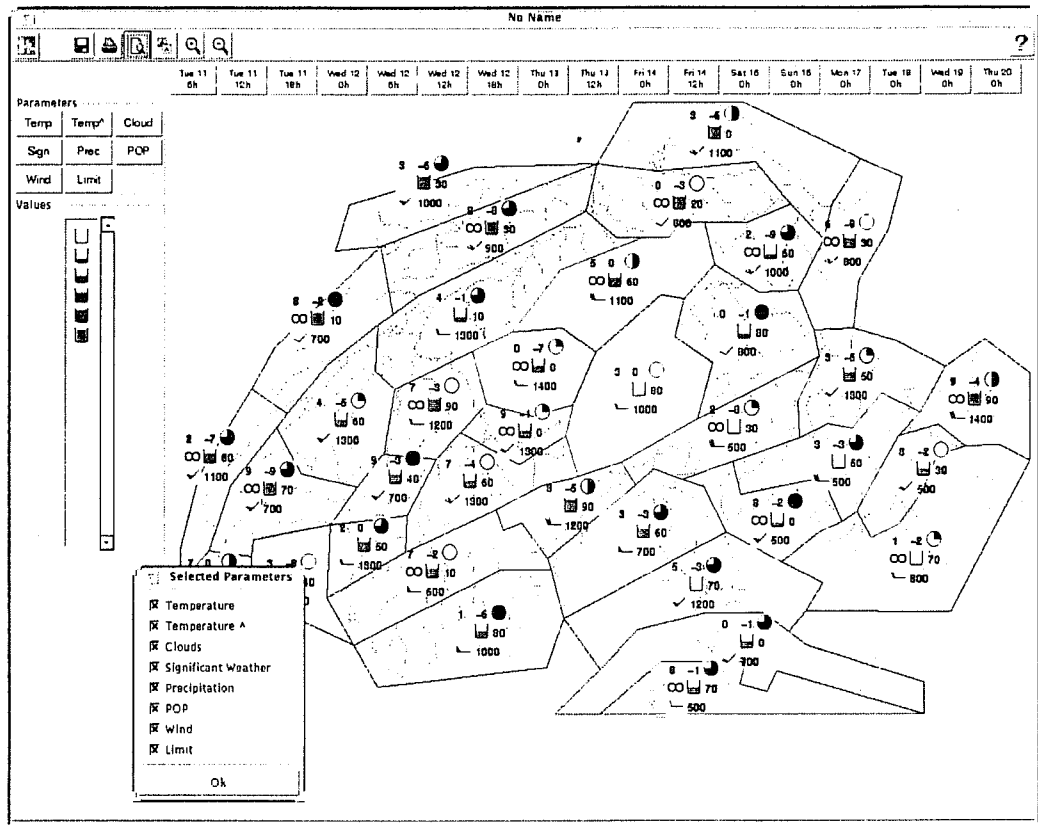
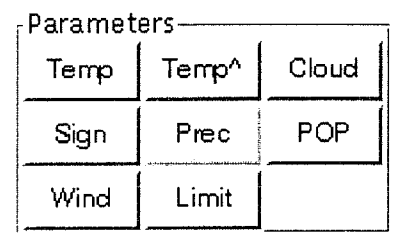


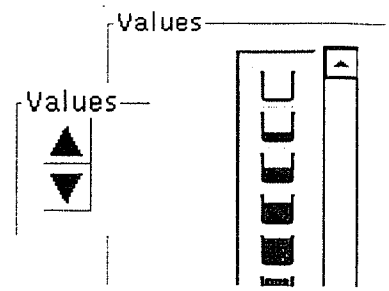
FIGURE 2. Forecast Editor Overview.

Regions. The background of this area is a map of Switzerland showing the main rivers and lakes but also the orography of the country. The map is divided into several forecast regions. The weather parameter values are arranged in a synop like order in the centre of each region. Only values of active (selected) parameters are displayed.
Regions may be selected or deselected just by clicking on them. It is possible to select multiple regions. Value editing is then active and can be done on all selected regions.

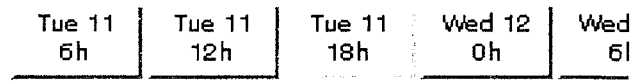
Parameters. This part contains buttons for all active parameters. The arrangement of the buttons is the same as the values and symbols in the region view.
Parameters may be selected by clicking on the buttons. Only one parameter may be selected at a time. If a parameter is selected the values panel shows the corresponding selectable values.



Values. There are two different types of values available: numbers and symbols e.g. temperature values are displayed as numbers and cloudiness values as symbols. Internally symbols are linked to a code number.
If a symbol is selected or an arrow button is pressed then the value(s) of the selected parameter and region(s) change.



Times. For each time interval there is a button. Only one time interval may be selected at a time. If a time interval is selected the corresponding values are displayed in the regions area.



Editing permissions. From the forecasting point of view Switzerland has three country parts (south, west, east), each with its own weather bureau (Locarno, Geneva, Zurich). Each bureau is responsible for its part of the forecast area. Thus a forecaster only has editing permission on fields of his own country part, i.e. it is not possible for a meteorologist from Zurich to edit regions form the south or the west.
To ensure consistency in the forecast over the three parts the daily conference phone calls between the weather offices may be used.

Timeseries representation. To follow the temporal sequence of the weather for a specific region we can choose the timeseries representation (meteogram). We click over a region and start the meteogram. This allows a control of the time consistency of the forecast. However we decided to restrict the editing capability to the geographical representation of a single time slice: we considered that an editing in both the time and space representation would set a high risk of inconsistency in the forecast. The complex terrain in Switzerland sets a bigger challenge in the geographical rather than in the temporal structure of the weather.

5 Implementation

5.1 Tools, Libraries, Hardware

- Programming Environment -- SNiFF+J 2.3.1 (TakeFive Software)
- Java -- JDK 1.1.3
- Libraries -- BWT 2.03 (KL Group)
- Operating System -- Solaris 2.4/2.5.1; Hardware -- SUN Sparc 4

5.2 Java Experiences

Profits

- Good public and commercial libraries (Java Developers Kit, GUI components)
- Object orientation, once you work with it you never want to go back
- Short development terms because of the easy reuse of formerly developed classes
- Well documented error messages makes debugging easy even without a debugger
- Uniform code documentation of all java libraries (javadoc)

Drawbacks

- Rather slow, still needs a fast machine to compile and run Java programs
- Fast market development: "The day you finish a program you can buy it!"

6 Present state and future developments

The present prototype was developed within our meteorological workstation project metAP. In this group both forecasters and computer scientists are represented and contribute to the design, development and tests of the different modules.

The introduction of the Forecast Editor will represent a major change in the whole forecasting process. A good acceptance of the tool is therefore essential. The user interface thus has to go through extensive tests and discussions among the forecaster community at the SMI.

Still under development are:
- Forecast DB (or forecast matrix)
- Connection between Forecast Editor and the Forecast DB
- DB ingest program for DMO/MOS data
- Data reduction programmes for the Forecast Product Editor (expert systems)
- Graphical module in the Forecast Product Editor
- Timeseries representation of the forecast (meteogram)

During '98 we will introduce step by step these different components to reach a complete functionality and support of the forecast-production process with the new tools.

7 Conclusions

The peculiarities of the climatic conditions over a very complex terrain like the Swiss Alps and the experience in weather forecasting in our country suggested an original solution in the rationalisation of the weather forecasting process. We believe that our proposal will guarantee an efficient, consistent and low cost method to produce forecast tailored to our customers needs.

The decision to use Java as a programming language is oriented towards the future and, even with a few temporary drawbacks, very promising.

Corresponding authors:

Daniel Matter (Java, Programming)
Swiss Meteorological Institute
Krähbülstr. 58
CH-8044 Zurich
mat@sma.ch

Paolo Ambrosetti (Philosophy, Meteorology)
Swiss Meteorological Institute
CH-6605 Locarno-Monti
pam@sma.ch