Proceedings of the 7th International Conference on Applied Informatics
Eger, Hungary, January 28–31, 2007. Vol. 2. pp. 327–333.

The investigation of the development of programming languages

Péter Takács, Zsolt Kristóf

University of Debrecen, Faculty of Health College
e-mail: {vtp, kristofzs}@de-efk.hu

Abstract

Our goal is to verify the categorization into generations from the time of publication of the programming languages. Our most important result is establishing a model which can give a more objective basis to the categorization into generations. To make this project we built a model based on the Kondratiev theory and we used modern statistical methods.

Keywords: history of programming languages, generations, Kondratiev theory, statistics, time series

1. Introduction

In a wider sense the development of programming languages began approximately 160 years ago with the work of Ada Lovelace. In fact the actual history of programming languages goes back to the mid 1940s. A lot of programming languages have come into existence during this period. Some languages spread worldwide, became popular and successful, on the other hand others operated in reduced circumstances. Most frequently the programming languages are classified into generations. Nowadays the languages are categorized into five or six generations [1, 2, 3]. Many times this categorization can be observed in the territories of education and research, mainly inside the domains of fundamental subjects.

Our goal is to verify the categorization into generations from the time of publication of the programming languages. Our most important result is establishing a model which can give a more objective basis to the categorization into generations. To make this project we built a model based on the Kondratiev Theory [4, 5] and we used modern statistical devices, such as the built-in trend analyzer tool of the *Statistica* [6] programme.

2. Building the model

In the course of model-building we applied the following components.

2.1. Sources

In the first step we deduced the publication time of the programming languages from the Wikipedia source which can be found on the Internet [7]. This source is widely known and detailed enough to make the necessary calculations based on it. The other indispensable component of this source is that on this page we can find not only the appearance of the programming languages, but their relations, too. We have to underline that the formed list can be edited freely by anyone on the Wikipedia page hence a lot of experts' opinions appear in this source.

2.2. Data processing

2.2.1. Evaluation of the languages

In the second step we processed our data in the forthcoming way. We assigned a numerical value to every programming languages. The numerical value of the point depends on the number of predecessor languages. We assigned 100 points to a fundamental language. These languages do not have any predecessors $(100/(1+0))$. The languages which have one predecessor language got 50 points $(100/(1+1))$. The point-value of the languages which have 2 predecessors turned up $33.3(100/(1+2))$. Generally the languages which have n predecessors got $(100/(1+n))$ point-values throughout the classification.

2.2.2. Annual summary

After the evaluation we summarized the point-values of the annual published programming languages. So essentially we got an evaluation which contained the annual development of the languages' published. It is demonstrated by Figure 1. We represented the summarized values on the "y" axis and the years on the "x" axis. In the next step we continued the examination of the annual values with statistical methods.

2.3. Statistical examination

In our further examination we transformed the data to the *Statistica* programme. Our goal was to prove the middle-range cycle. We used two methods for this.

2.3.1. First method – The adaptation of the Kondratiev Model

In the first method we adopted the Kondratiev theory to our data [5]. The main points of this theory are the following:

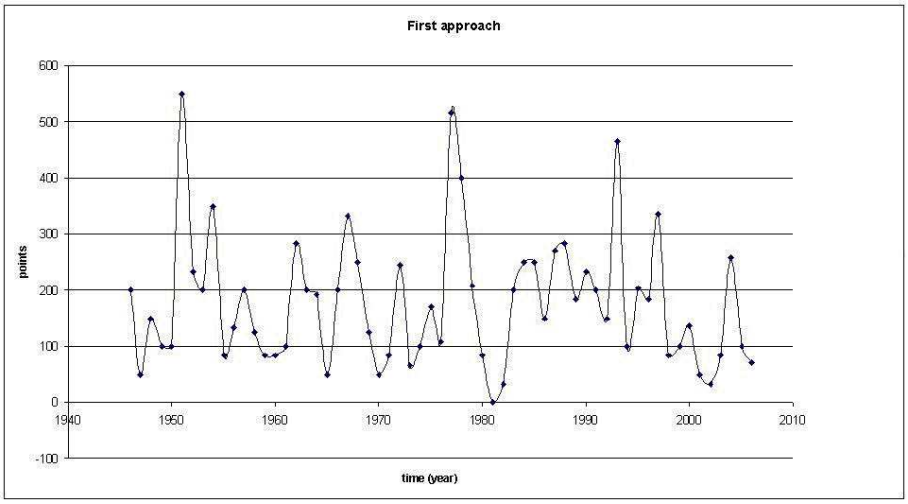

Figure 1: Summarized point-values by year

In the first step we determined a long-range tendency using a linear trend. This means fitting a lineal in our model ($\hat{y}_i$). Because our purpose was the demonstration of the middle-range cycle we shifted out the long range tendency from data ($y_i - \hat{y}_i$, y_i the summarized point-value). In this case according to the theory the middle-range and the short-range trends were left over. We realized the smoothing of the short-range trends by a five-point moving average transformation. Following this way we got the middle-range cycle which is illustrated by Figure 2.

2.3.2. Second method - Using the T4253 H(x) Filter

In the second method we used the built-in T4253 H(x) filter[1] in the *Statistical* programme for the original annual numerical values. We got the graph which can be seen in Figure 3 as a result.

We cross-checked the result from the Kondratiev model with the result of the T4253 H(x) filter. It could be observed that these two cycles' periodicity was very similar. We signed the peaks with numbers (1-2-3-4-5) for the further examination.

[1]**T4253 H(x) Filter.** "This transformation consists of several passes of moving average/median smoothing, and is a powerful filter for smoothing a series. ... following transformations are performed: (1) a 4 points moving median centered by a moving median of 2, (2) a 5 point moving median, (3) a 3 point moving median, and (4) a 3-point weighted moving average using Hanning weights (.25, .5, .25), (5) residuals are computed by subtracting the transformed series from the original series, (6) steps 1 through 4 are then repeated for the residuals, (7) the transformed residuals are added to the transformed series. In practice, this filtering method often produces a smooth series while maintaining the salient characteristics of the original series." [8]

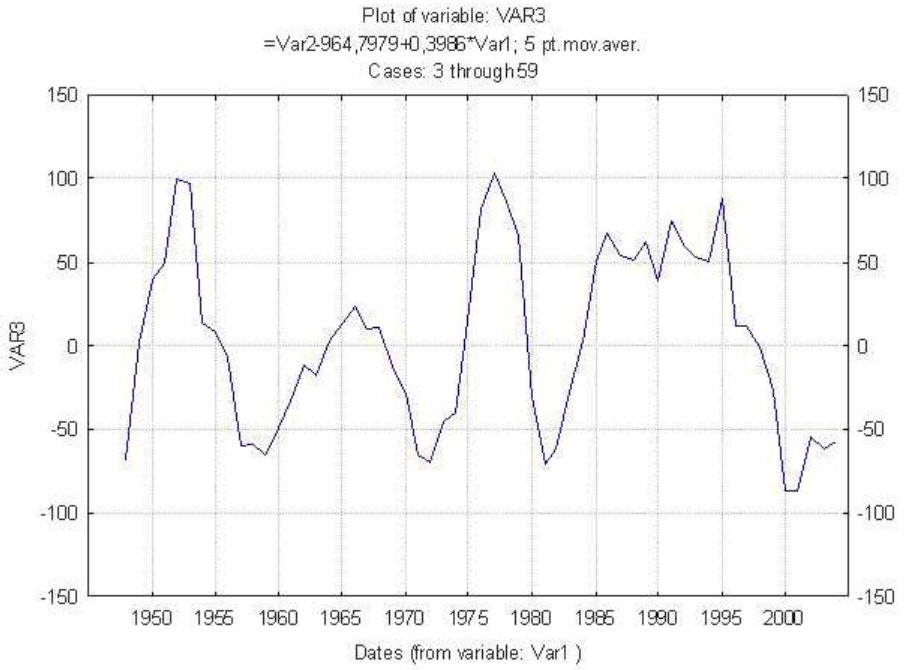

Figure 2: Application of the Kondratiev theory

3. Evaluation of the results - analysis

The peaks which can be seen in the Figure 3. are the following ones:
1. peak $\sim$ 1952, 2. peak $\sim$ 1967, 3. peak $\sim$ 1977, 4. peak $\sim$ 1986, 5. peak $\sim$ 1995.

If we want to define the peaks, there are a lot of ways ahead of us. Some of the sources separate 5 or 6 generations [1, 3, 9]. Another approach highlights those languages which have had the most intense impact on the development. Comparing with the peaks that we received, the different professional sources show quite a mixed picture.

Usually the first peak covers the first and the second generations.

The second and third peaks can be interpreted as the third generations. Perhaps here we can differenciate the high level languages (third peak) and the structured programming (fourth peak). In both cases it is typical that the languages of the most intense impact can be found between the peaks. *Fortran*, *Algol*, *Cobol* lie between the first and second peaks and *Pascal*, *C*, *Prolog* lie between the second and third peaks.

The fourth peak shows quite a mixed picture. We observed that the languages

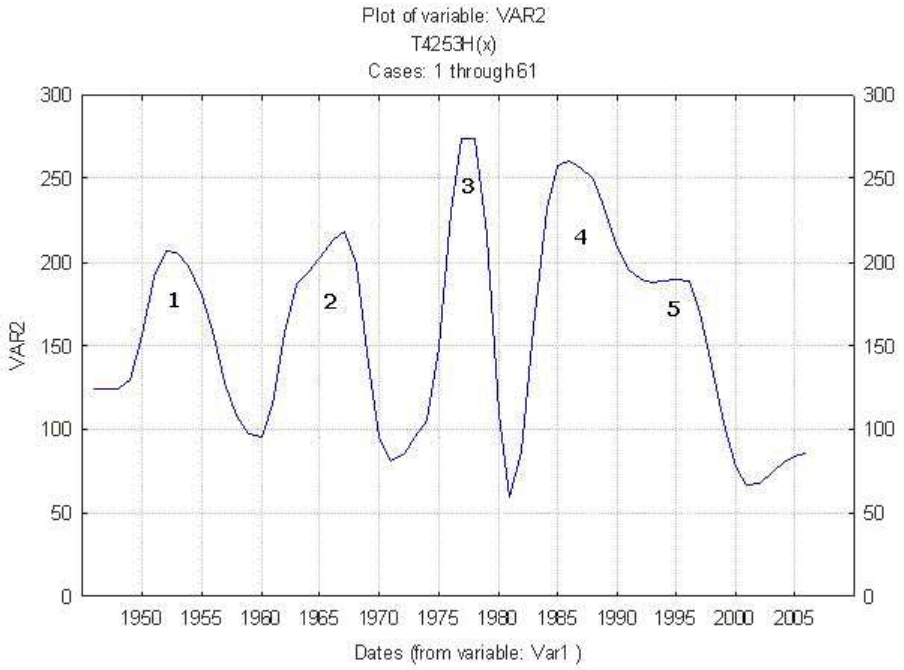

Figure 3: Application of the T4253 H(x) filter

with the most intense impact lie in a valley between 1979 and 1984 (*dBase II*, *Smalltalk*, *ADA*, *C++*, *SML*). They run up after 1984. Howewer according to the generational classification these languages belong to several generations.

The fifth peak is the era of the programming languages used at industrial level (*PHP*, *Delphi*, *Java*). It is interesting to note that at the fourth and fifth peaks the previously observed tendency (ascending-descending periods) breaks. After the fourth peak, approximately in 1992 a descending period was expected to come. Instead some languages at industrial level appeared, thus forming the fifth peak. After this peak another descending period could be observed in about 2001.

The graph has an interesting feature around the year 2000, where it reaches the bottom similar to the one in 1981. At that time the arrival of 4th generation languages gave it a boost. In these days this upthrust has not appeared yet but we think a new orientation can come into view in the near future which may cause an upswing that we perceived in the previous years.

4. Limits of the model

The model can still be improved from different aspects. For example, the assigning of the numerical values, the configuration of the scale, the predecessors of the languages and the depth of their relations can be a view-point of further investigation. The linear trend is just one approximate form.

Several unconsidered factors can improve this model, such as the evolution of the hardwares and their cyclical development. We can use additional explanatory elements like the cyclical behaviour of the personal life, the scientific work and research. We could not find similar approaches in the scientific literature (professional papers).

5. Conclusions

In our paper we managed to create a newer, perhaps a more realistic model concerning the development of programming languages. During our research we could observe that the professional sources could be understood in many different ways. Roger Clarke's statement is true in any case:

"The concept of generations of programming languages and application software technology has been popular, but has lacked concrete and useful definitions." [11]

We believe that we will obtain a more exact approach by defining and improving a more accurate model and we hope that our work will inspire other researches on this topic.

References

[1] Wikipedia, Category: Programming language classification, `http://en.wikipedia.org/wiki/Category:Programming\_language\_classification` Visited: 2007.05.14.

[2] Wikipedia, Programming language, `http://en.wikipedia.org/wiki/Programming\_languages` Visited: 2007.05.14.

[3] Cynbe ru Taren, Six Generations of Programming Languages, `http://www.muq.org/~cynbe/ml/ml.html` Visited: 2007.05.14.

[4] Wikipedia, Kondratiev wave, `http://en.wikipedia.org/wiki/Kondratieff\_Cycle` Visited: 2007.05.14.

[5] Hunyadi, L., Mundruczó, Gy., Vita, L., Statisztika, *Aula Kiadó*, (1996), 7. fejezet, Idősorok elemzése, 525–605.

[6] Statistica, Statistica Homepage, `http://www.statsoft.com/` Visited: 2007.05.14.

[7] Wikipedia, Timeline of programming languages, `http://en.wikipedia.org/wiki/Timeline\_of\_programming\_languages` Visited: 2007.05.14.

[8] Statistica, Programme Help, Version 7.1.

[9] Campbell, J. G., Lessons in Object-oriented Programming and C++, `http://www.jgcampbell.com/oopcpp99/html/node4.html` Visited: 2007.05.14.

[10] A Chronology of Influential Computer Languages, `http://people.ku.edu/~nkinners/LangList/Extras/famous.htm` Visited: 2007.05.14.

[11] Clarke, R., A Contingency Approach to the Application Software Generations, `http://www.anu.edu.au/people/Roger.Clarke/SOS/SwareGenns.html` Visited: 2007.05.14.