## KEEP A CLEAN NOSE, WATCH THE PLAINCLOTHES

# DO YOU NEED A WEATHERMAN?

*By Claude Bédard*

**John Chandioux may not be Météo's sole parent, but he's certainly its guardian angel. Météo, for those not steeped in MT lore, is Canada's government-backed project for the machine translation of weather bulletins, operational since 1977.**

**John Chandioux, now 40, has devoted most of his adult life to machine translation. And in that arcane yet passion-filled world, Chandioux has earned a reputation both as an independent-minded entrepreneur and as an outspoken systems developer with a passion for real-life performance.**

**LT: How did you first get interested in machine translation?**

When I was a child. My father was a French army translator, and in the early 60s, we lived in the United States. This was MT's *belle époque* – or you might say its *années folles*. Anyway, my father was head of the French translation bureau at the Pentagon, when the military got interested in Systran.

Then I went to university in Grenoble, France, where I studied applied linguistics. There, I met Bernard Vauquois, head of the Groupe d'Etudes pour la Traduction Automatique (GETA) project. After I graduated in 1973, Bernard helped me get a job on the Traduction Automatique de l'Université de Montréal (TAUM) project here in Canada.

**LT: What was happening at TAUM in 1973?**

They were just releasing a prototype system – with no applications in sight. Then, in 1974, a weather bulletin translator from Environment Canada happened to remark to someone that his job was so boring it might just as well be done by a machine. Anyway, we took Environment Canada up on the idea, and in the same year, we started work on a weather bulletin project, which we called Météo.

The Météo prototype was delivered in the spring of 1976. But for some reason – probably because most of the people at TAUM were basically into theoretical research – no one was particularly interested in making it fully operational. They just put in on a shelf and let it gather dust. Anyway, in 1977, the government sponsored TAUM for another – more ambitious – project: Aviation. The aim here was to translate 90 million words of aviation maintenance manuals from English into French.

**LT: Was that when TAUM and you started to fall out?**

You could say that. I had my own ideas about how to tackle Aviation. I wanted to start with modest, real-life goals, by taking the emphasis off theoretical problems. I was more interested in whatever could help the translator be more productive – that is deal with the practical problems of how to maximize productivity on a particular type of text.

Most of the other Aviation team members couldn't resist going for perfect output – and on a translation model influenced by a very competent but rather purist translator, André Petit. That's why the Aviation prototype turned out either practically perfect translations or nothing at all.

As far as linguistic strategies go, I found their transfer stage much too heavy. My preference was to perform only syntax-based transformations at the transfer stage, with any lexical-based transformations at the synthesis level.

But my biggest complaint at TAUM-Aviation was about management. There were no master specifications for the project and no real leadership. No one knew who the boss was. Everybody did what they wanted.

**LT: So you resigned?**

Right. And I took Météo with me. In the spring of 1977,

I teamed up with Benoît Thouin and got a contract from the Canadian government's Translation Bureau to implement the Météo prototype.

That's when the shock came – the system could only translate 40% of the text. There were two problems. First, the corpus used by TAUM was not representative of the full range of weather conditions, because it was too seasonally oriented – winter weather conditions, for instance, had not been accounted for. Second, the translation of weather bulletins had since been extended to all the regions of Canada. We found some very significant variations in the style of the original English.

On top of all of this, TAUM's software, programmed in Systèmes-Q, proved undependable for a production environment. So we had to rewrite it in a "crash-proof" format.

Météo became operational in 1977, which is when I turned my mind to a personal project that had been on my mind for some time – namely, to write a more dependable programming metalanguage than Systèmes-Q. Alain Colmerauer, who invented Systèmes-Q and later Prolog, encouraged me to do it on a microcomputer – remember this was back in 1977 – because he believed that the future of MT was tied to the smaller machines. I called my metalanguage GramR.

**LT: How long did it take to develop it?**

From 1977 till 1981. When I started out, I had absolutely no experience in computer programming – quite a shortcoming for someone who wanted to create a programming language! So first I had to teach myself programming. I earned my living doing maintenance on the Météo system, translating computer documentation, and reselling microcomputers.

When GramR was ready, I did a feasibility study for the Translation Bureau, demonstrating that microcomputer-based MT was viable. This came at just the right time, too. One year later, in 1983, the Translation Bureau was forced to get rid of its Control Data computer. They were going to have to change it either for a Cray – which would have entailed porting the program to the new machine – or a microcomputer.

Riding on the success of my feasibility study, I built a complete Météo system in GramR on a 68000 machine with 512 Kb of RAM – a big micro at the time – and proposed a trial comparing it with the existing system. After using both systems in parallel for several months, the translators concluded that Météo 2 – the name of the GramR system – did the job less expensively, practically as fast, and more reliably than the Control Data system.

So, beginning in October 1984, the Translation Bureau signed a contract with me to rent a turnkey Météo 2 system – and it's been operational ever since.

**LT: What's so great about GramR?**

GramR is part of a family of languages originating in Grenoble. But what's unique about it is that it incorporates all the functions of MT in one language. At Grenoble, they needed three: ATEF (the parser), GETA (the tree transducer) and SIGMOR (the generator). On top of that, GramR runs on microcomputers.

One thing I like about GramR it its essentially deterministic nature. Systèmes-Q was a parallel analysis tool – rather too difficult to control. I don't think that so much parallelism is necessary to produce good machine translation. It's a very interesting development tool, but for actual production I'm not so sure.

**LT: Not even for general-purpose systems, where the source text is not highly predictable?**

I'm still not convinced that a deterministic approach would fail, provided you don't go strictly left-to-right or right-to-left. My approach is to start by analyzing the more solid constituents in order to build islands of certainty, and then to analyze whatever lies around them while retaining the possibility of reconsidering certain decisions.

In fact, you can have a totally deterministic strategy and still allow for reinterpretation, which is like doing parallelism without paying the price for it. Whenever you build on something uncertain, you keep a trace of it in order to take it apart if necessary.

**LT: What about third generation MT systems?**

I don't have much faith in them, at least for the foreseeable future. Any concrete AI is limited to subdomains. And there's a big, and mundane, barrier of complexity as soon as you get out of less than strictly limited domains.

**LT: What have you been doing since 1984?**

I've been looking for other MT applications, while building a freelance translation system. I haven't been able to convince anyone to participate in a joint venture for another system like Météo, so I've been turning to more modest spin-offs of MT technology, such as spelling and grammar checkers, and an automatic accentuator for telexes – all in French.

I've also created EasyDOS, a French and English natural language interface for DOS users, which is selling very well. These products have the immediate advantage of being consumer-oriented and can be sold at a profit. But my heart still really lies in MT, and I'm just waiting for an opportunity to get back into it.

**LT: What would you do if some big rich backer were to come along right now with a walletful of venture capital?**

What would interest me most right now would be a serious offer to build a dedicated machine translation system more ambitious than Météo – with more difficult linguistic problems and bigger dictionaries – in a domain where the volume and demand both exist. I'd much prefer this to building a general-purpose system, whatever the financial or technical means available.

Such a system wouldn't have to be revolutionary, but whatever it did, it would have to do right – a second-generation system, no idiom tricks, an emphasis on well made dictionaries, and no AI.

**LT: Something like what you wanted TAUM-Aviation to be?**

You got it.

*LT's Canadian correspondent, Claude Bédard is a technical translator and MT consultant based in Montreal, Canada.*

PHOTO BY CLAUDE BEDARD

**John Chandioux: When I started out, I had absolutely no experience in computer programming – quite a shortcoming for someone who wanted to create a programming language!**