

# Open Meeting on the Next LP Language

---

Bart Demoen, Catholic University of Leuven  
Peter Stuckey, University of Melbourne

Manuel Hermenegildo, Tech. U. of Madrid / U. of New Mexico

- Context: a number of efforts (some ALP-backed). E.g.:
  - The ISO-Prolog standard
  - “Grass roots” Prolog standardization effort
  - A number of self-declared “next generation” systems ( $\lambda$ -Prolog, HAL, Ciao, ...)

And now:

- The “New Logic Programming Language” effort (NLPL).
- NLPL is a clear mission from the ALP!

## The “grass roots” Prolog standardization effort

---

- Objective is developing standardized:
  - public-domain library,
  - foreign interfaces,
  - syntactic extensions,
  - possibly predicate description approach,
  - possibly modules,
  - etc.

(Coordinated by Jan Wielemaker/Bart Demoen/Manuel H.)

- Advantage: most Prolog developers and “classical experts” involved.
- Disadvantage: somewhat chaotic, progress very irregular, ...
- Worth continuing:
  - Traditional Prolog systems are still our real ‘business card’!
  - Libraries etc. may be useful for TNPL.

## NLPL – The Mission

---

- NLPL is quite different:
  - It is about *language* design.
  - Idea is to reflect best design ideas and implementation strategies that we have beyond Prolog after many years of research.
  - Result should achieve a broad consensus in the community.
  - E.g., such that, given a decent implementation, will likely be used in papers, newsgroups, and teaching.
  - Potential to one day grow into an even more credible LP language for industrial/scientific use than Prolog.

## NLPL – This Meeting

---

- Discuss the language features which we consider essential to the language
- Brainstorming! New ideas!
- Try to set up an action committee for designing and eventually building the next logic programming language  
*A committed group which will pursue the refinement of the design, in order to implement the language.*

## Some possible characteristics

---

- The classics: DCGs, delays, library, ...
- Constraints, being able to add new constraint domains
- Feature terms (records)
- Functions, higher-order (lambda Prolog?)
- Tabling
- Types and other declarations: modes, assertions in general.
- Automatic checking of assertions
- Objects/inheritance
- Concurrency-threads/distributed execution/parallelism
- Interfaces to other languages
- Other control rules: Andorra, breadth first, ...
- Autodocumentation

## Some possible characteristics (Contd.)

---

- Modules (which?)
  - Code expansion facilities
  - Pure subset available (or pure overall?) / declarative I/O
  - Persistence
  - (Constructive) negation
  - Answer set programming,
  - Abduction, non-monotonic reasoning
  - Inductive LP
  - Programming environment, ...
- 
- Integrating what we know (e.g., Prolog/CLP w/tabling, constraints, Andorra and parallelism, ...) already a *huge* challenge...
  - It is clear that discussion of these issues needs more than just this one meeting...

## Other questions

---

- Why a Next LP language? Is Prolog not good enough?
- Is this a putsh by Ciao? By HAL? By... :-)
- Is the implementation technology ready for this?
- Who will own the eventual implementation? (GNU...)
- What time frame are we looking at?
- Is it the intention to start from scratch?
- Is this an attempt to kill Prolog?
- Should it be backwards compatible?
- Is there funding for the implementation effort?
  
- Voice your opinions!