

LOP is Language for Languages?



Alexey Efimov 1,410 posts since

Aug 21, 2002

Hello,

I just read article, it contains previous articles from Sergey's blog. This is really new layer of programming concept, but I also look some disadvantages:

1. You can't add languages into your MPS for everything, what you can make on OS. Again restrictions.
2. You will always be restricted by OS libraries and virtual machines.
3. Previously MPS orientation on graph model is better, because it is strongest for math definition. So, if you allow programmer to do 'unrestriction action', you will lose concept on whole MPS system.

I definitely think that a new programming system must have different kind of ideas. Some of AI instead of manual coding. All that may do programmer - program AI itself. In this case delay for coding can be reduced. I think that MPS can't reduce it as it shows on your picture. Programmer still makes hand work, so it is not a step towards 'reducing coding delay'.

What do you think about AI for inspections, intentions and code writing depends on hints?

Thanks!

LOP is Language for Languages?



[Sergey Dmitriev](#) 263 posts since

Oct 12, 2001 1. **Re: LOP is Language for Languages?** Nov 17, 2004 6:58 PM

>>1. *You can't add languages into your MPS for everything, what you can made on OS. Again restrictions.*

Could you please be more specific and provide an example of language that can't be added to MPS?

>>2. *You will always restricted by OS libraries and virtual machines.*

Again, it is not clear what kind of restriction do you mean here? In NPS you usually shouldn't use OS API directly, but you can generate code, that uses any OS APIs and libraries. So, this approach should provide even more portability then Java. For example we can have the same source and then generate code for JVM and for .Net from it.

>>3. *Previously MPS orientation on graph model is better, because is strongest for math definition. So, if you allow programmer to do 'unrestriction action', you will loose concept on whole MPS system.*

First, MPS models are based on graph representation now, so no changes were made in this area.

Second, what is "unrestriction action" - I don't understand?

May be you mean that there was some runtime model, described in my early postings to Russian weblog, that allowed to interpret MPS models directly? I just see that interpreting semantic is only one of ways how to define semantic of a program/language, another way is - the definition of generator. MPS will support both variants in future, though now we are concetrated on generative approach, because it is much more practical for current time - it allows easily integrate programs, written in MPS with programs, written in other languages/platforms.



[Alexey Efimov](#) 1,410 posts since

Aug 21, 2002 2. **Re: LOP is Language for Languages?** Nov 17, 2004 7:09 PM

LOP is Language for Languages?

Could you please be more specific and provide an example of language that can't be added to MPS?

Sure, i not talk about such unavailability. I just say that you maybe add one-two languages, but then you will make something more and you will need complex refactoring over all languages, i thing that MPS will bottle neck. Maybe i'm wrong.

>>2. You will always restricted by OS libraries and virtual machines.

Again, it is not clear what kind of restriction do you mean here?

Hm.. In your article, word 'freedom' equals to 'portability'? I want to say, that programmer always restricted by OS, and no one system can't solve it really.

Second, what is "unrestriction action" - I don't understand?

Ability to add languages as wish programmer, for every action every programmer may build own language and that is it?. I thing it no good, IMHO.

Sergey, i'm interest by some new level of programming concept, and your atricle is very interest for me. But i also think, that Generation concept must be replaced by AI for really new concept. Or is it wrong way?

Thanks!

LOP is Language for Languages?



[Rob Harwood](#) 896 posts since

Sep 10, 2002 3. Re: **LOP is Language for Languages?** Nov 18, 2004 1:16 PM

I just say that you maybe add one-two languages, but then you will make something more and you will need complex refactoring over all languages, i thing that MPS will bottle neck.

The 'complex refactoring' that you talk about sounds to me like 'a higher level language with transformations'. For instance, you might want to have in your program: A GUI language, a collection language, a database language, a web application language, a special business-oriented language (like maybe accounting rules or something). And then you think, "I just want to write my program at a higher level, like Customers, Accounts, Transactions, User Interface Tables, etc.). So, you create a higher-level language that ties all of these lower-level DSLs together. You write transformations to translate the user interface into web pages and/or GUI, Accounts and Customers map to the database language, Transactions map to SQL-like query language, and everything is glued together using the collection language.

I want to say, that programmer always restricted by OS, and no one system can't solve it realy.

Of course, if the OS can't do something, then the programmer can't do it either. That's given.

Hmm, I wonder what an OS written in MPS would be like?

But i also think, that Generation concept must be replaced by AI for realy new concept. Or is it wrong way?

Sergey talked to me about the *next* next paradigm, which will probably involve AI (at least, more goal-oriented style of programming, not necessarily a full AI). However, this is very far away probably. We need a bridge from OOP to AI, which LOP will provide.

LOP is Language for Languages?

Rob Harwood



[Stephen Cryan](#) 1 posts since

Nov 17, 2004 4. **Re: LOP is Language for Languages?** Nov 17, 2004 7:57 PM

Hello,

I've tried a few approaches for generative programming, and I'm thinking about investing the effort of trying this one.

Here's what I've tried:

1. javacc - too complicated for me, but it does seem to provide maximum flexibility
2. java emitter templates (from eclipse) - much simpler, but I ended up with messy templates.

Has anyone else tried these - or any other - tools. I'd be interested in hearing about it.

thanks

Steve



[Alex Roytman](#) 179 posts since

Aug 21, 2002 5. **Re: LOP is Language for Languages?** Nov 18, 2004 12:19 AM

I have pretty detailed object model for to reverse engineer relational databases and to generate java code from it (mainly JDO-backed objectmodel and required metadata). I use velocity for templates.

All my projects have 100% generated persistence layer because I support merge of generated code with hand edited code in generated classes on any "named" construct level

LOP is Language for Languages?

(class, method, field, nested class) I use Sun's Java parser directly. I was thinking to add support for UML now instead **I would love to try to make an O/R mapping LOP out of it**



[Rob Harwood](#) 896 posts since

Sep 10, 2002 6. Re: LOP is Language for Languages? Nov 18, 2004 1:21 PM

There's a very cool new tool called Ruby on Rails (<http://www.rubyonrails.org/>) which provides a nearly seamless mapping to a database, eliminating most of the work for most of the projects. It uses the Active Record pattern, but my main interest is the power of Ruby to support this as almost a DSL-inside-Ruby.

Until MPS goes EAP, you might want to check out Ruby as well.



[Alexey Verkhovsky](#) 1 posts since

Nov 22, 2004 7. Re: LOP is Language for Languages? Nov 23, 2004 12:05 AM

Actually, if you follow changes in Rails CVS, it is sort of evolving into a set of domain languages, exploiting Ruby's ability to alter behaviors at runtime.

Schema definition language of ActiveRecord was there from beginning (stuff like has :messages; belongs_to :user etc), but now similar constructs are being added in other layers.

By the way, the "Collection Language" of Ruby is an awful lot nicer than the examples in the article 😊