Envisioning Future Language Workbenches

voelter@acm.org
www.voelter.de
@markusvoelter
Markus Völter
itemis voelter { ingenieurbüro für softwaretechnologie //

Where I come from

1

Precision

Formulas, Rules
Data Structures
Tables
Values

+ Tests

!=

Performance
Scalability
Robustness
Deployment

Programming

All languages shown in this talk

are built with the open source JetBrains MPS language workbench.

MPS Language Workbench

Language Workbench

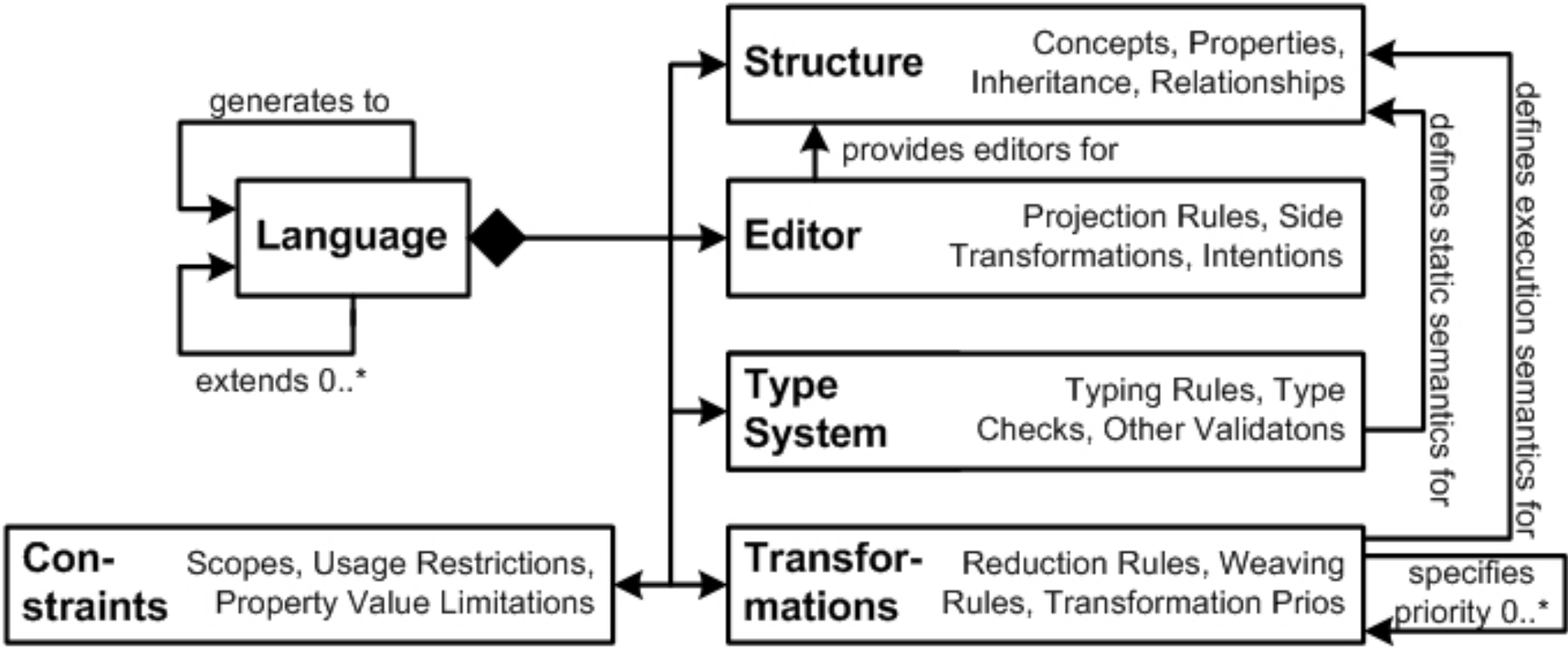
Open Source, by Jetbrains

Very Expressive

Used for years in industry

Vast Experience

MPS Language Workbench



+ Refactorings, Find Usages, Syntax Coloring, Debugging, ...

berbwvekFF (lkm_akt_param; lkm_faell_param; ber_zweck_param; kz_rzw_param)

Funktionenmodell berbwvekFF

Formale Beschreibung

Funktion: berbwvekFF
Programmquelle: vmsctfa1.c
Produkt-Typ: Fonds **PK-Typ:** Kapital-Konto
Status: 18.1

Parameter-Attribute

lkm_akt_param
lkm_faell_param
ber_zweck_param
kz_rzw_param

Verwendete VADM-Attribute

Keine verwendeten VADM-Attribute, werden automatisch hinzugefügt

Rückgabe-Attribut

bwvek

aufgerufene Funktionen

VTRKermbtgfaellFF (a)
berbweinzelFF (a; b; c)

Beschreibung

Die Funktion liefert den Barwert per @lkm_akt_param des vorschüssigen Zahlungss
@lkm_akt_param bis @lkm_faell_param – jeweils einschließlich. Zahlungszeitpunkt
#lkm_akt_param – 1# bis #lkm_faell_param – 1#. Der Parameter @kz_rzw_param steu
Zahlweise des Zahlungsstroms. Möglich sind zur Zeit nur die Ausprägungen 0 (Zahl
und 12 (monatliche Zahlungsweise).

Hilfsvariablen

kz_bf_hilf

Verarbeitungen

```
Schleife über lkm_faell_hilf = lkm_akt_param bis lkm_faell_param
  Falls kz_rzw_param = 12
      kz_bf_hilf = 1
    sonst
      kz_bf_hilf = VTRKermbtgfaellFF (lkm_faell_hilf)
  Ende Falls kz_rzw_param = 12
  bwvek = bwvek + kz_bf_hilf * berbweinzelFF (lkm_akt_param; lkm_faell_hilf - 1
Ende Schleife über lkm_akt_param bis lkm_faell_param

return bwvek
```

Formale Beschreibung

Funktion: berbwvekFF
Programmquelle: vmsctfa1.c
Produkt-Typ: FONDS **PK-Typ:** KAPITAL-KONTO

	Name	Verw.	Entität
verwendete Attribute:	lkm_akt_param	E	PARAMETER
	lkm_faell_param	E	PARAMETER
	ber_zweck_param	E	PARAMETER
	kz_rzw_param	E	PARAMETER
	bwvek	A	RETURN

aufgerufene Funktionen: berbweinzelFF
VTRKermbtgfaellFF

Verarbeitungen

Die Funktion liefert den Barwert per lkm_akt_param des vorschüssigen Zahlungsstroms der Höhe 1 von Monat lkm_akt_param bis lkm_faell_param – jeweils einschließlich. Zahlungszeitpunkte sind jeweils die Monatsbeginne, also **lkm_akt_param** -1 bis **lkm_faell_param** – 1.

Der Parameter **kz_rzw_param** steuert die zu berücksichtigende Zahlweise des Zahlungsstroms. Möglich sind zur Zeit nur die Ausprägungen 0 (Zahlungen zu den Beitragsfälligkeiten) und 12 (monatliche Zahlungsweise).

Schleife über lkm_faell_hilf = **lkm_akt_param** bis **lkm_faell_param**

Falls **kz_rzw_param** = 12

kz_bf_hilf = 1

sonst

kz_bf_hilf = *VTRKermbtgfaellFF*(lkm_faell_hilf)

Ende Falls **kz_rzw_param** = 12

bwvek = **bwvek**

+ kz_bf_hilf * *berbweinzelFF*(**lkm_akt_param**, lkm_faell_hilf - 1, **ber_zweck_param**)

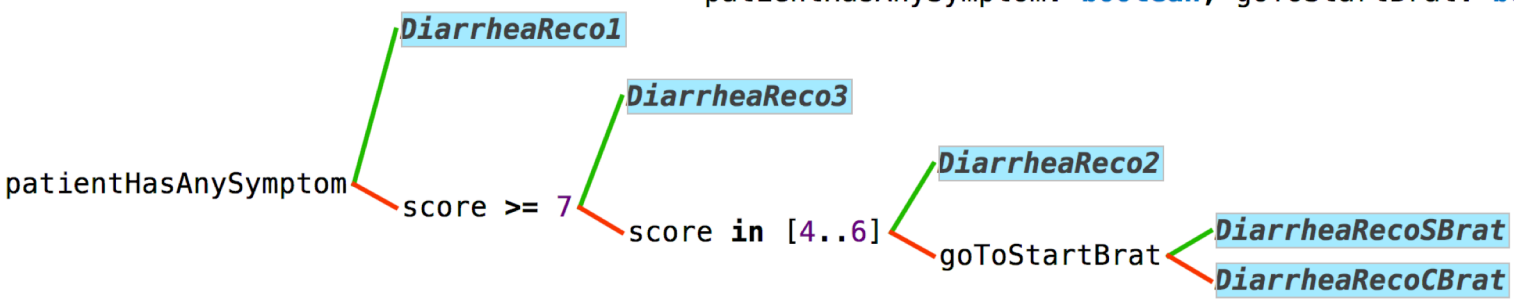
Ende Schleife

return **bwvek**

```
decision table BpScoreDecisionTable(sys: bpRange, dia: bpRange) =
```

		dia					
		<= 50	[51..90]	[91..95]	[96..100]	[101..109]	>= 110
sys	<= 90	1	1	3	4	5	6
	[91..140]	2	2	3	4	5	6
	[141..150]	3	3	3	4	5	6
	[151..160]	4	4	4	4	5	6
	[161..179]	5	5	5	5	5	6
	>= 180	6	6	6	6	6	6

```
decision tree DiarrheaStoolsDecisionTree(score: DiarrheaStoolsOverBaseline,
                                         patientHasAnySymptom: boolean, goToStartBrat: boolean)
```




```
type temperature: number[36|42]{1}
type measuredTemp: number[35|43]{2}

val T_measured: measuredTemp = 42.22
val T_calibrated: temperature = T_measured * 0.93
```

Error: type number[32.55|39.99]{4} is not a subtype of number[36|42]{1}

Name: UVG-Leistungen

Unterhaltsvorschuss

Zeitangabe: laufend
Häufigkeit: monatlich einmal
Leistungskontext:
Leistungsart: Leer
Zählart: uvg
Anspruch Beginn: Anfang - Unbegrenzt: junger Mensch.geburtsdatum

Anspruch Ende: 01.01.1800 - 31.12.9999 **: min(junger Mensch.**geburtsda
12 Jahre

Zeitraum für Berechnung: Anfang - Unbegrenzt: {standardzeitraum, standardz

zweckgebundene Leistung: ☐
dem Grunde nach: ☐

Zeitraumbezogene Daten

nullwerte Anzeigen : **boolean** = 01.01.1800 - 31.05.2016 **: true**
01.06.2016 **- Unbegrenzt: false**

berechnungsart : **berechnungsarttyp** = 01.01.1800 - 31.12.9999 **:**

Bezugsobjekte: << ... >>
Attribute:
bemerkung : **string** wird validiert
antragsdatum : **Datum**

Nebenberechnungen

Name: Kindergeld für vollen Monat

(01.01.1800 - 31.12.9999)

Rechnungsart: wenn: wird geboren **mit junger Mensch als** person **dann voller**
sonst: taggenau
Begünstigtenprinzip: ☐
monatswert = Kindergeld 1. Kind

zwischenergebnisse = [<< ... >>]
endergebnis = monatswert

```
1  isore rest rule TimingPattern_07_ObjectsAcrossRows_Modified_TandA_897_1Iteration {
2    description: 1381R3.ARCHD.0609.TimingPatternLanguage_TestScenarioMap.xlsx

3    parameters: TimePeriodObjectTypA4

     patterns:
4
5          1          2          3          4          5          6          7
6
7    scenario                                  < TimePeriodObjectTypA4 >
8    scenario   TimePeriodObjectTypA1 >                                TimePeriodObjectTypA6
9    scenario                        < TimePeriodSpecifier2::Duration = 24 Hours >
10   scenario                                  < TimePeriodSpecifier3::Duration = 15 Minutes >
11   scenario                                                         △ TimeSpikeObjectTypA5

     }
```

database databaseOneElementAcrossRows

Type	Begin	End	Duration	Occurence
eTimePeriodObjectTypA	500	550	50	

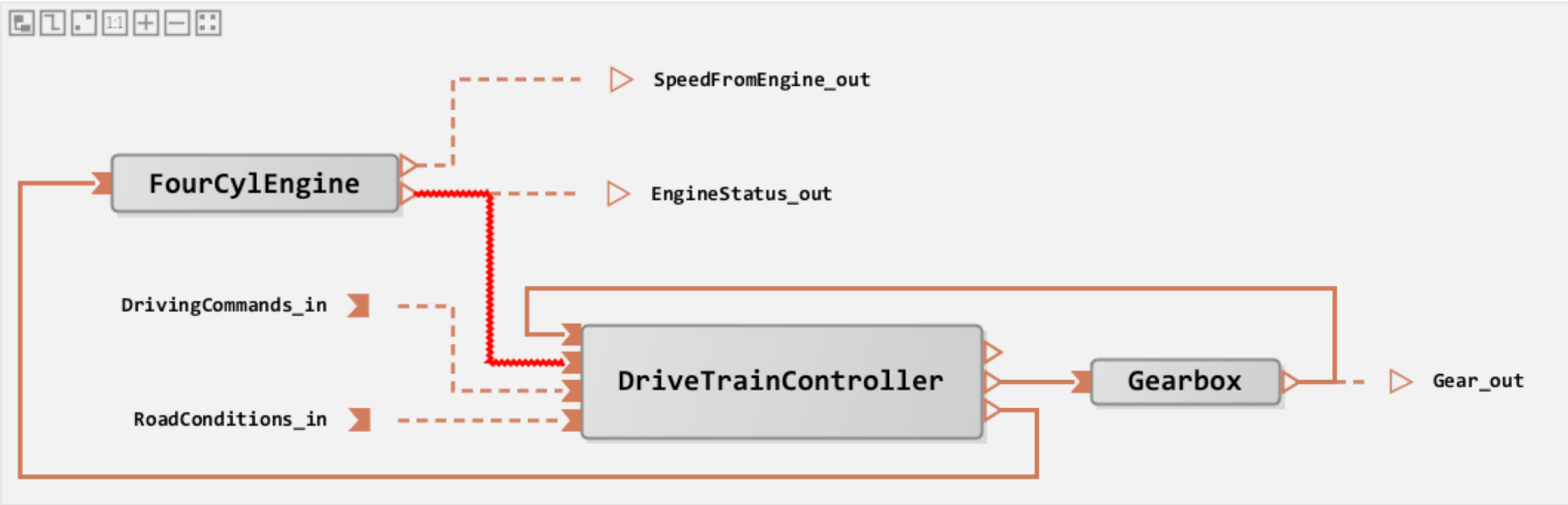
database databaseOneAndMoreIterationsHappy

Type	Begin	End	Duration	Occurence
eTimePeriodObjectTypA	50	100	50	
eTimeSpikeObjectTypA				86000
eTimePeriodObjectTypA	86020	86030	10	

```
risk factor Ausrüstung with levels
  Standard                   : Gängige IT-Ausstattung (z.B. Notebook, frei verfügbare OBD-Diagnose-Tools)          = 0
  Spezialausrüstung          : Professionelles Werkstattequipment (CAN-Cards, Diagnoseequipment)                   = 4
  Maßanfertigung             : Mind. ein Sonderwerkzeug (z.B. nur für Hersteller verfügbares Werkzeug oder         = 7
                               Kostenintensive Ausrüstung >50.000 € (z.B. Elektronenmikroskop))
  Mehrfache Maßanfertigungen: Mehrere Sonderwerkzeuge                                                              = 9

risk factor Kenntnis des Zielobjektes with levels
  Öffentlich :   = 0
  Intern     :   = 3
  Vertraulich:   = 7
  Geheim     :   = 11


required attack potentials
  Sehr gering   = 0
  Gering        = 10
  Moderat       = 14
  Hoch          = 20
  Mehr als hoch = 25
```

Attack Potentials Table		Required consecutive attack potential				
		Mehr als hoch	Hoch	Moderat	Gering	Sehr gering
Required initial attack potential	Mehr als hoch	Mehr als hoch	Mehr als hoch	Mehr als hoch	Mehr als hoch	Mehr als hoch
	Hoch	–	Hoch	Hoch	Moderat	Moderat
	Moderat	–	–	Moderat	Gering	Gering
	Gering	–	–	–	Gering	Sehr gering
	Sehr gering	–	–	–	–	Sehr gering

```
public functional component DriveTrain {
  produces SpeedFromEngine
  produces EngineStatus
  produces Gear where it < gearsCount
  consumes RoadConditions
  param int gearsCount
  consumes DrivingCommands
```




```
}
```

```
vector<int16, 3> aVector = [1; 2; 3] * 512;

vector<int16, 3> resultOfCrossProduct = aVector x aVector;

matrix<int16, 2x3> aMatrix = [1 + 2   2 * 7   42; 3   51   24];

matrix<int16, 3x2> transposedMatrix = aMatrix^T;
```

```
int32 averageIntArray(int32[] arr, int32 size) {
```

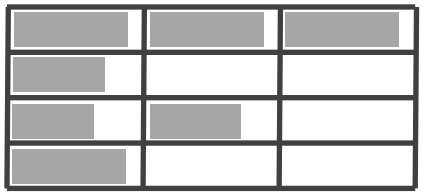
$$\text{return } \frac{\sum_{i=0}^{size} arr[i]}{size};$$

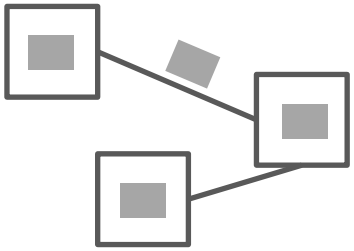
```
} averageIntArray (function)
```

```
val beitragProzentsatzArbeitnehmer: %%% = 1.50%
val beitragProzentsatzArbeitgeber: %%% = 1.50%


daten ArbeitslosenversicherungStamm {
  beitragsgruppe            : arbeitslosenversicherungBeitragsgruppe
  unternehmenRechtskreisOst : boolean
}


ergebnis [monatlich] ArbeitslosenversicherungErgebnis {
  arbeitgeberBeitrag  : €€€
  arbeitnehmerBeitrag : €€€
}


fun getSvBruttoGekürzt(rechtskreisOst: boolean, svBrutto: €€€): €€€
  =
```

rechtskreisOst	svBrutto > bbgOst	svBrutto > bbgWest	wert: €€€
true	true		bbgOst
false		true	bbgWest
			svBrutto

MPS: Notational Freedom

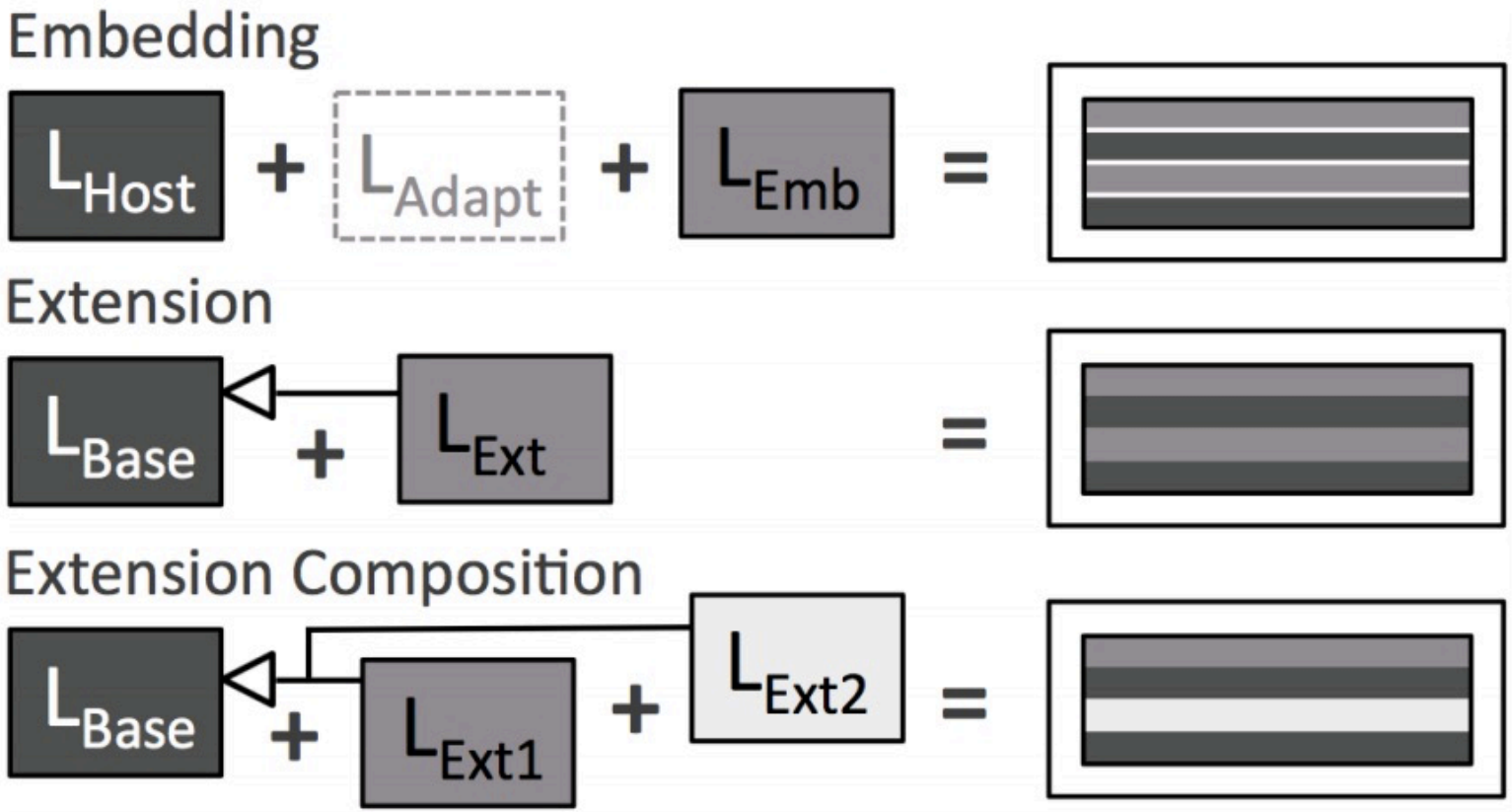




100% crucial for acceptance in domain!

Modeling is more than diagrams!

MPS: Language Composition

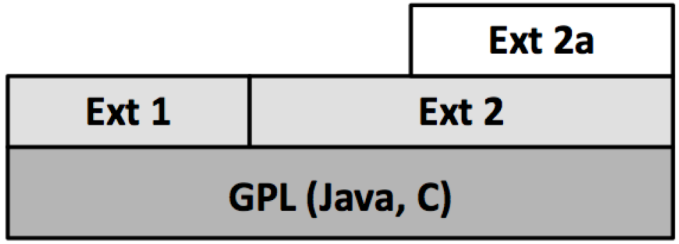


SPLE on Language Level!

Language Extension Patterns

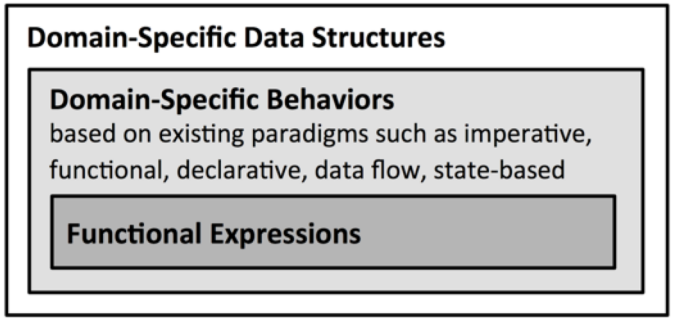
GPL Extension

Reuse GPL incl. Expressions and TS
Add/Embed DS-extensions
Compatible notational style
Reduce to GPL



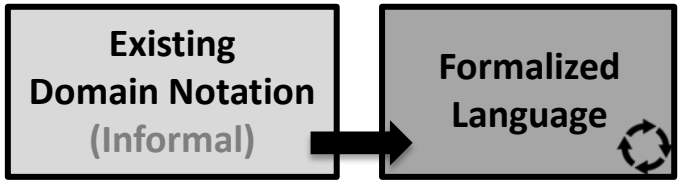
New Language

Analyze Domain to find Abstractions
Define suitable, new notations
Rely on existing behavioral paradigm
Reuse standard expression language
Interpret/Generate to one or more GPLs

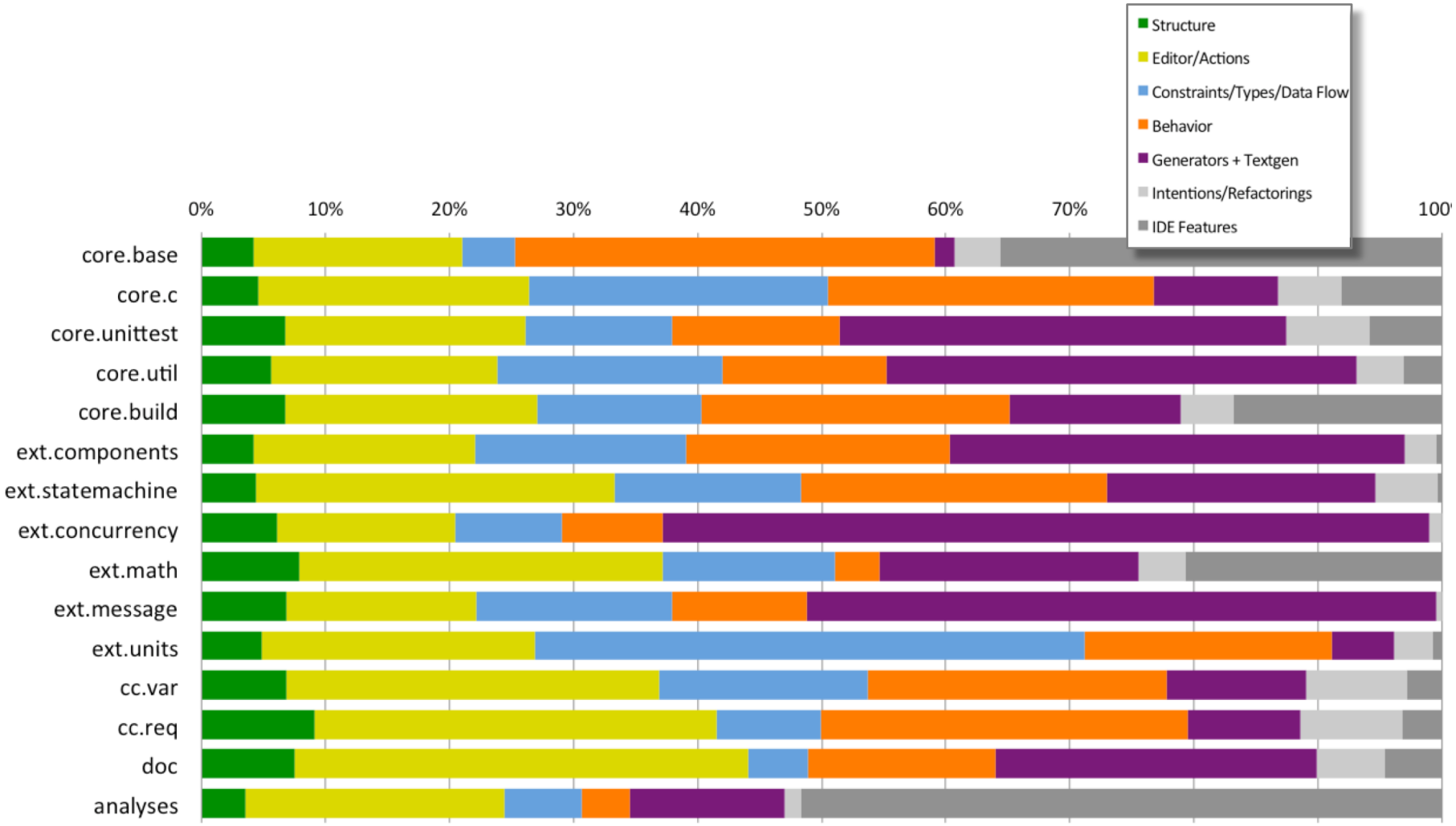


Formalization

Use existing notation from domain
Clean up and formalize
Generate/Interpret
Often import existing „models“



Language Flavours



Language Concerns

II

1

Realtime Incremental Transformations

Example Use Cases

- Mapping of system models to model checkers for interactive verification of temporal properties
- Flattening of component hierarchies for type checking
- Flattening of hierarchical feature models for path expression evaluation
- Weaving in of safety concerns into C code.
- Desugaring of business DSLs to a core functional language; that core language would then feature an interpreter, a compiler and an integration with an SMT solver

What is Shadow Models?

Functional **transformation language** with support for fixpoints

Incremental execution upon change of input model

Unidirectional, but with first-class support for **lifting** results

Fully integrated into **MPS** IDE

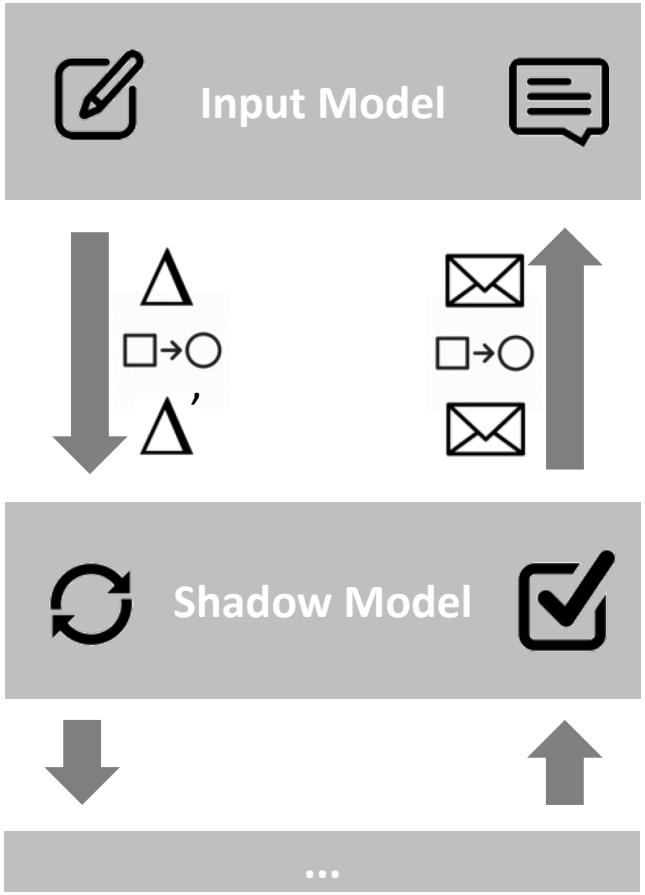
Code: https://github.com/JetBrains/MPS-extensions/tree/master/code/shadowmodels

Docs: https://jetbrains.github.io/MPS-extensions/extensions/shadowmodels/

Mechanics

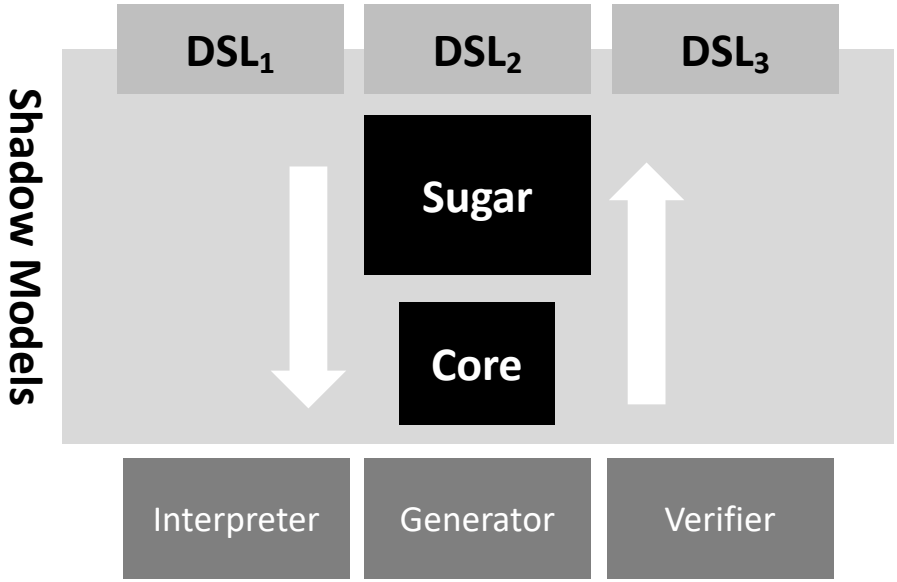
Might be stacked

- User **edits** the input model
- A **delta** is propagated into the transformation engine
- A **change** on the shadow model is produced
- This **triggers** analysis or update of results in Shadow Model
- Results/messages are **lifted** back to input level
- Results are **annotated** to the input model



Kf2

- Minimal, Expressive Core
  - Functional
  - Reactive
- Sugar on top
- Interpter, Generator and Verifier below
- Realtime-Trafo with Shadow Models



Kf2

TaxExample

```
module TaxExample {

  enum states { BW }

  calculation root = sub
    item sub = 10


}
```

TaxExample

Warning: the node is in a transient model. Your changes won't be saved.

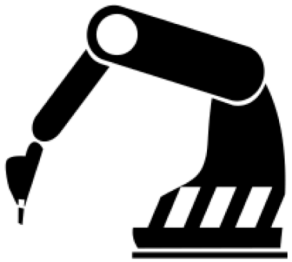
Warning: the node is in a read-only model. Your changes won't be saved

```
module TaxExample {

  val states_BW = -2084976718

  record root_Env {
    root : int
    sub  : int
  }
  fun root_root(values: root_Env) : int = {
    if values.root != 9999
      then values.root
      else root_sub(values)
  }
  fun root_sub(values: root_Env) : int = {
    if values.sub != 9999
      then values.sub
      else 10
  }


}
```

2

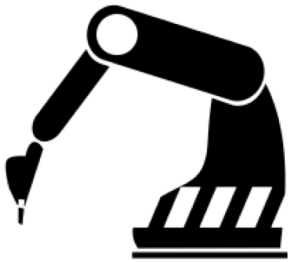
Semantics

LWBs



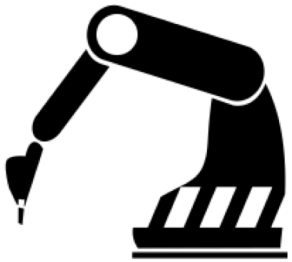
Semantics

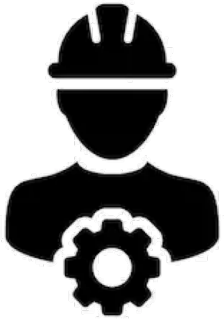
LWBs



Semantics

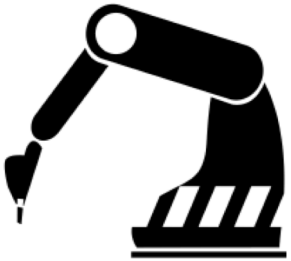
LWBs

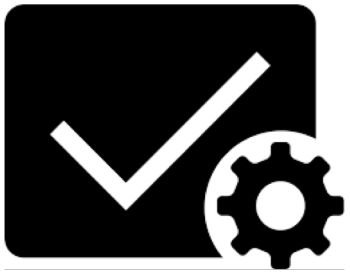




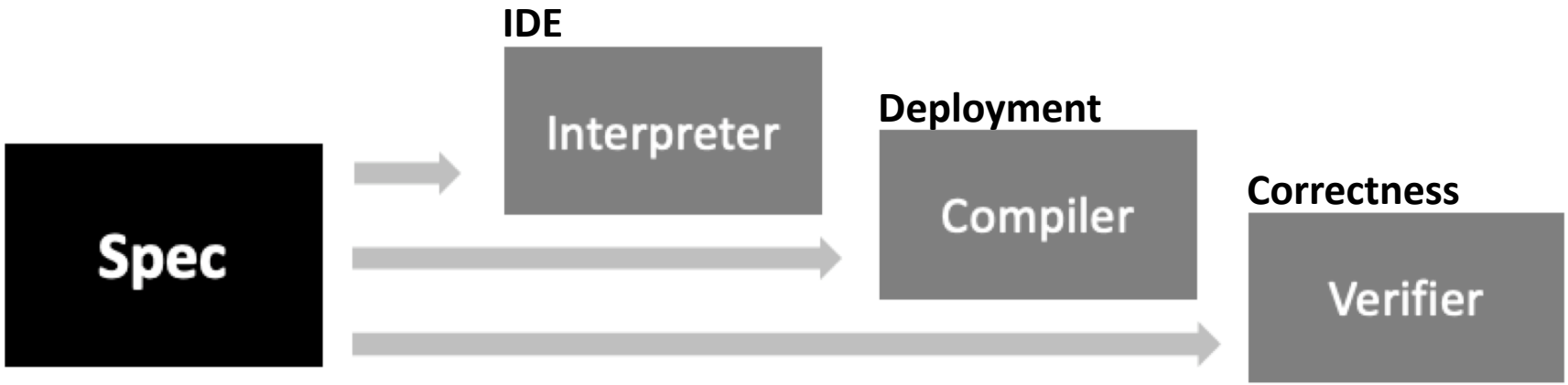
Semantics

LWBs + Semantics





Semantics



Functional + Imperative + Reactive

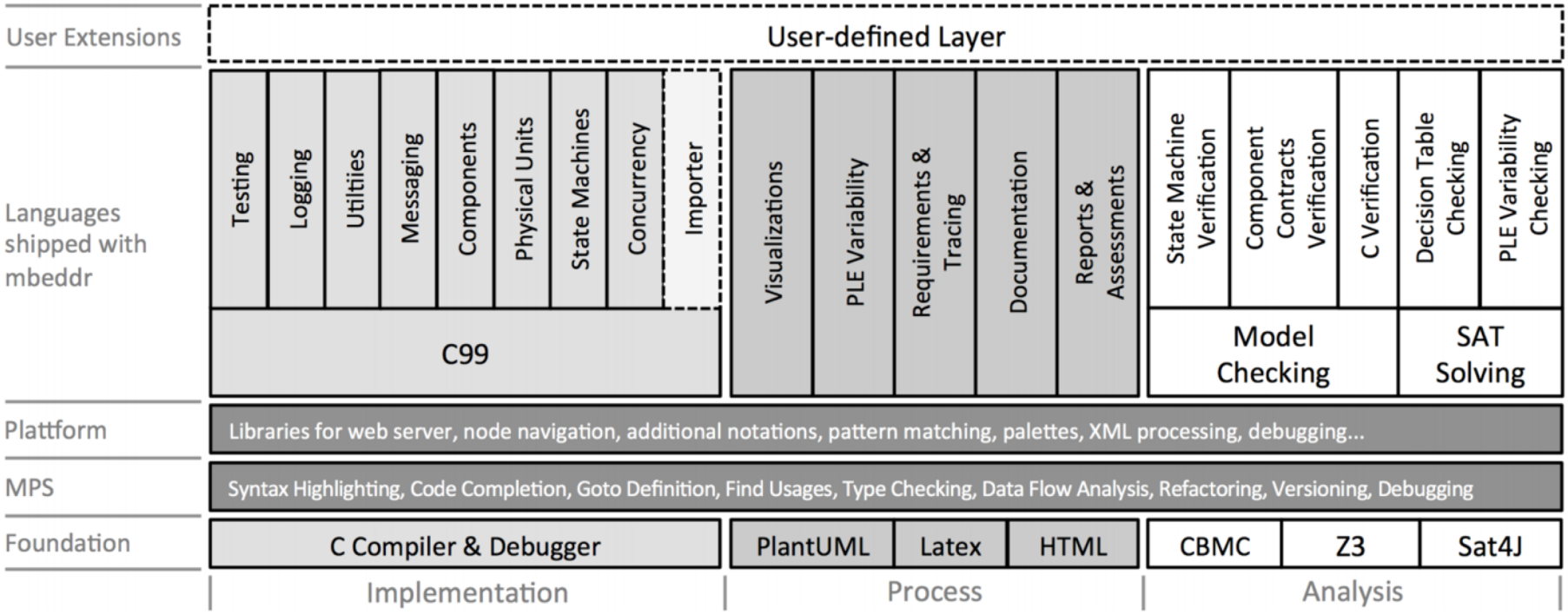
The tools exist: K, Funcons, VDM

But there is very limited integration with LWBs.

Exception: Spoofax + Funcons

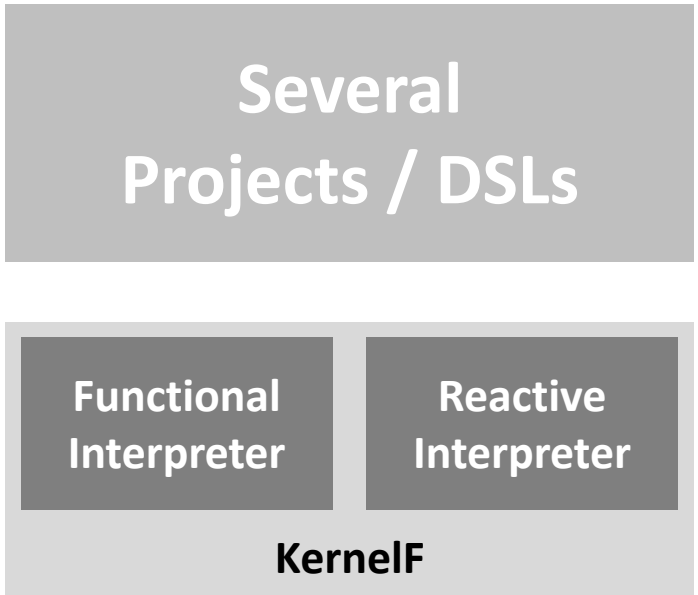
Semantics

Grow on (imperative) C



Semantics

Grow on Functional Language...



Semantics

Grow on Functional Language...

... and other Foundations.

Several Projects / DSLs

Lots of ideas ☺

Functional Interpreter

Reactive Interpreter

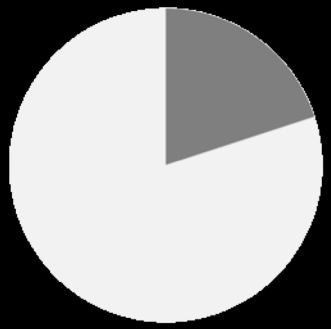
KernelF

Solver

Model Checker

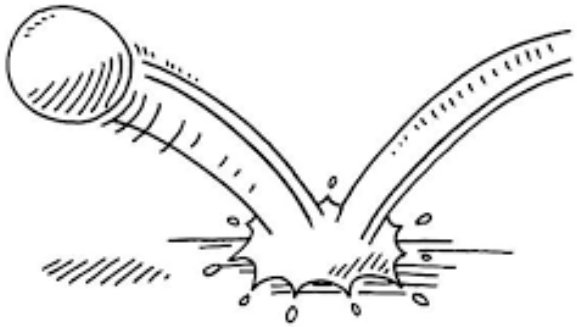
Rules Engine

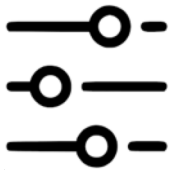
3



Liveness

Beyond Balls and Trees





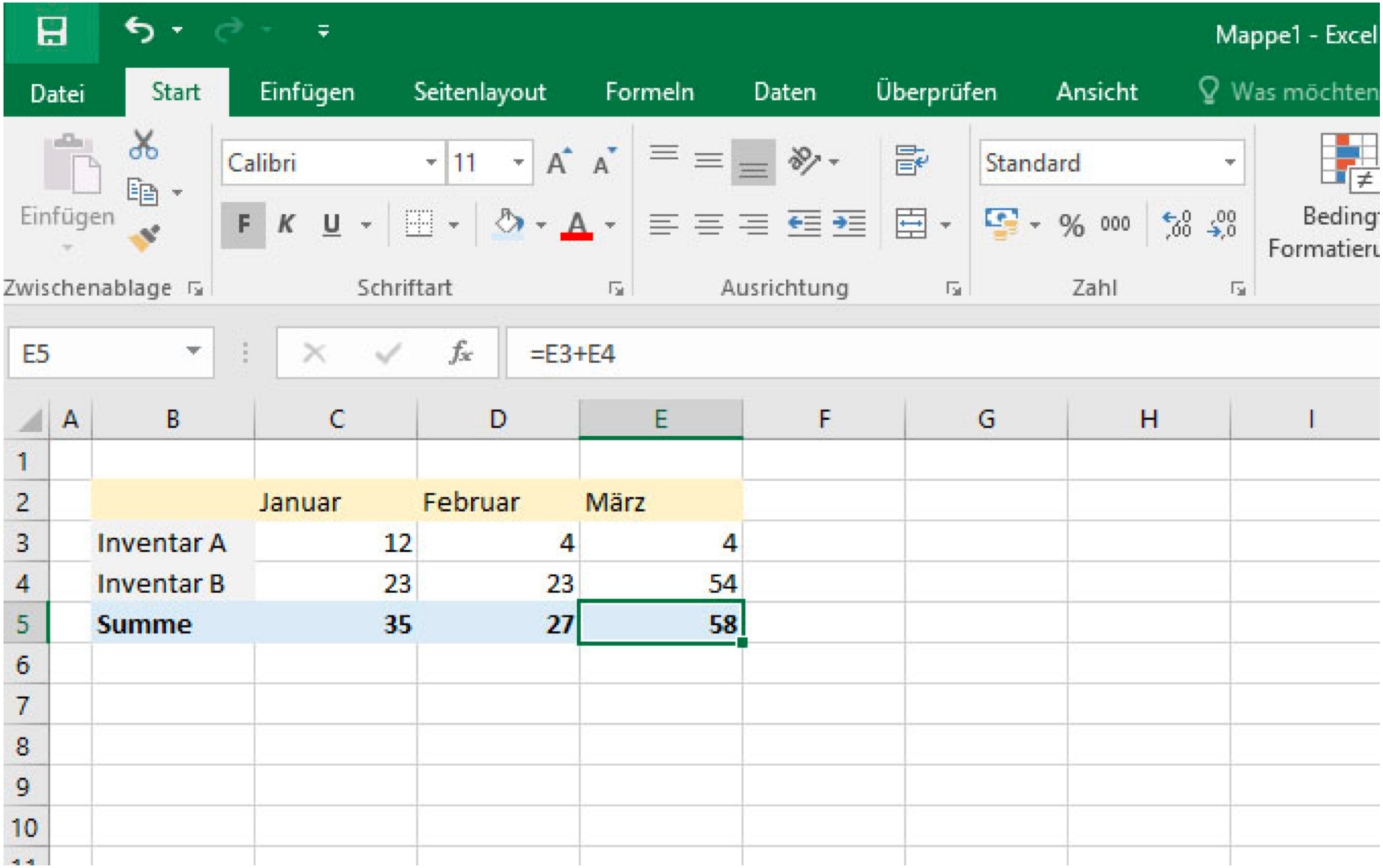
Not too much has happened since the legandary demo.

Overlay program execution data
Run tests and illustrate results in the background.
Instance-based programming (Jonathan Edwards)

That‘s it?

What else can we do?
Which LWB-support can be build?

Why do people like Excel?



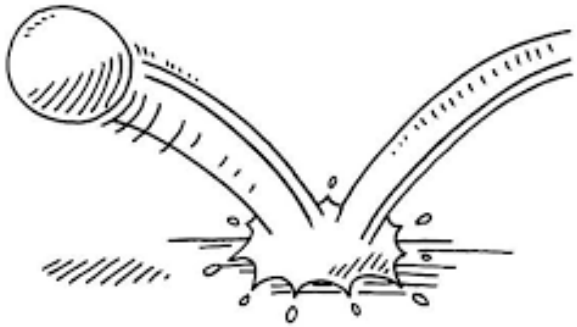
Beyond Balls and Trees

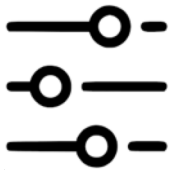
```
interpreted<false> module TaxExample {

  enum states { BW, BY, MV, BE }
  fun isEasternState(s: states) = s == MV || s == BE

  calculation incomeTax = alt [ salary < min => 0 EUR
                                otherwise    => salary * percentage / 100 ]
    container item Citizen
      item married : bool
      item salary : money
      item state : states
    item min : money = if married
                       then 2000 EUR
                       else 1000 EUR
    item percentage =
```

	married	otherwise
isEasternState(state)	20	40
otherwise	10	30

```
  assert equals run incomeTax with [ married false
                                     salary 5000 EUR
                                     state  BW ]  == 1500

  assert equals run incomeTax with [ married true
                                     salary 5000 EUR
                                     state  MV ]  == 1000

  assert equals run incomeTax with [ married false
                                     salary 500 EUR
                                     state  MV ]  == 0

  assert equals run incomeTax with [ married true
                                     salary 1500 EUR
                                     state  BW ]  == 0

}
```

Beyond Balls and Trees





Not too much has happened since the legandary demo.

Overlay program execution data
Run tests and illustrate results in the background.
Instance-based programming (Jonathan Edwards)

That‘s it?

What else can we do?
Which LWB-support can be build?

4

Debuggers

Debuggers

Different ones for different language paradigms

- Functional => Tree
- Imperative => Stepping
- Declarative => ?

Trace Explorer assert { val v = getR().addr.first if v.street.length < 3 then "LessThan3" else "Bigger" }

- **assert** assert { val v = getR().addr.first if v.street.length < 3 then "LessThan3" else "Bigger" } [AssertTestIte
  - **actual:** { val v = getR().addr.first if v.street.length < 3 then "LessThan3" else "Bigger" } ⇒ **Bigger** : Strin
    - **val** v ⇒ **#Address[Oetz, #City[70327, St ...** : RecordValue (4 ms) [ValExpression]
    - if v.street.length < 3 then "LessThan3" else "Bigger" ⇒ **Bigger** : String [IfExpression]
      - **cond:** v.street.length < 3 ⇒ **false** : Boolean (1 ms) [LessExpression]
      - **else:** Bigger ⇒ **Bigger** : String [StringLiteral]
  - **expected:** getOetz() ⇒ **Oetz** : String [FunctionCall]

```
if v → #Address[Oetz, #City[70327, Stuttgart]] .street → Oetz .length → 4 < 3 then "LessThen3" else "Bigger"
   → Oetz
   → 4
   → false
   → Bigger
```

Debuggers

Different ones for different language paradigms

- Functional => Tree
- Imperative => Stepping
- Declarative => ?

Often the (semantic reverse) of generation

it has to recover DSL semantics for watches, views, breakpoints

Related to simulators and „model animators“

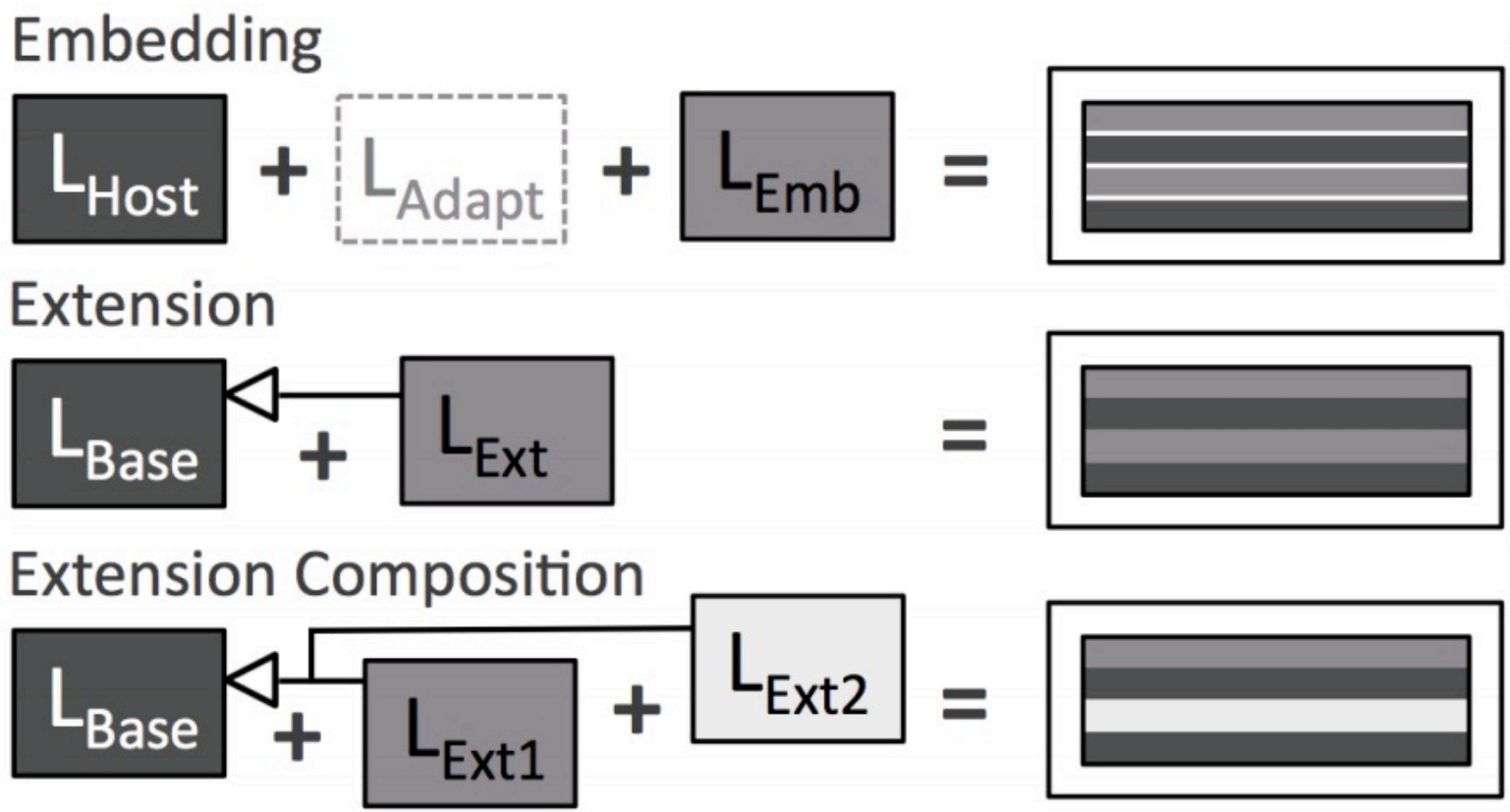
Realtime vs. Post Mortem

No good tool support for building them!

5

Language Parametrization & Interfaces

Language Composition



Language Composition

Defining „abstract“ languages with holes

Example: State machines with pluggable expression language

What is the type of those that can be plugged in?

Language Types or Interfaces / Megamodels

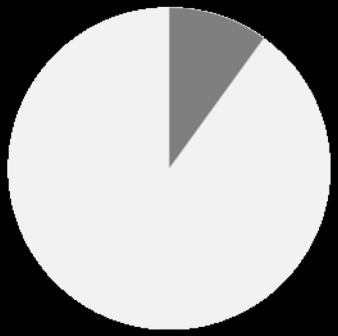
Example: How do you descibe a „valid“ expression language?

Does this include execution sematics?

Tool Concerns

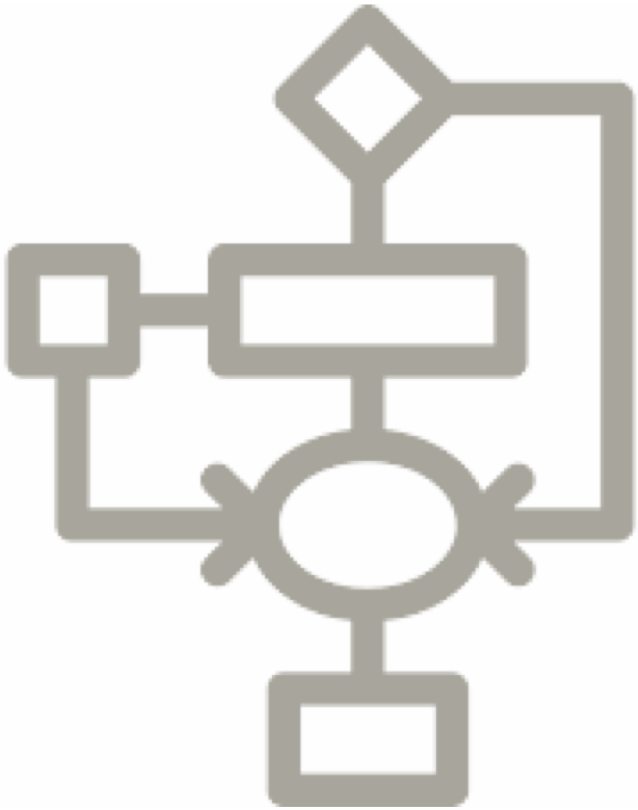
III

6

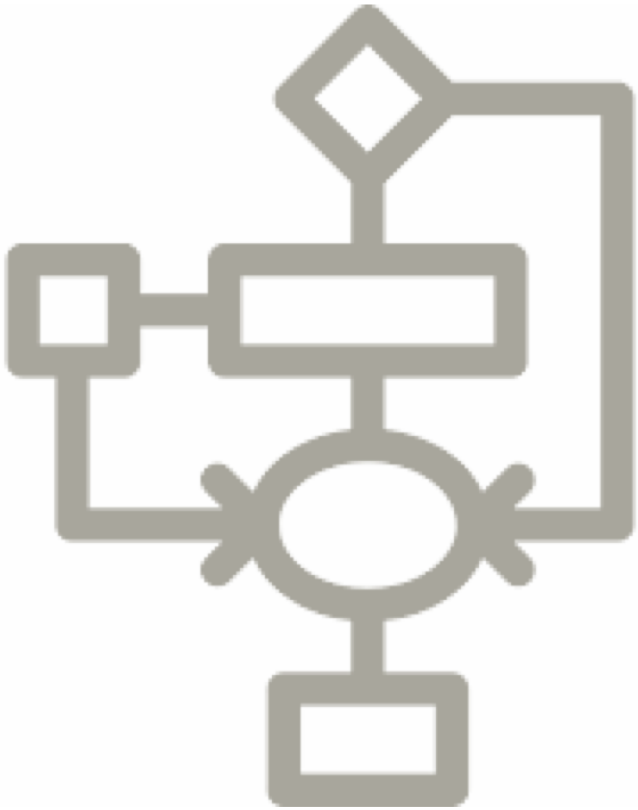


Guidance

Applications

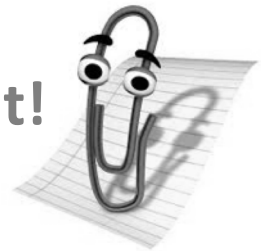


Applications vs. Modeling Tool



Novice Users need Guidance!

But not like that!



Applications vs. Modeling Tool

Novice Users need Guidance!

Applications vs. Modeling Tool

Novice Users need Guidance!

What am I working on?

Roles

Applications vs. Modeling Tool

Novice Users need Guidance!

What am I working on?	**Roles**
What should I do next?	**Tasks, Workflow**

Applications vs. Modeling Tool

Novice Users need Guidance!

What am I working on?	**Roles**
What should I do next?	**Tasks, Workflow**
How complete is my model?	**Model States**

Applications vs. Modeling Tool

Novice Users need Guidance!

What am I working on?	Roles
What should I do next?	Tasks, Workflow
How complete is my model?	Model States
What is the quality of my model?	Metrics

Applications vs. Modeling Tool

Novice Users need Guidance!

What am I working on?	Roles
What should I do next?	Tasks, Workflow
How complete is my model?	Model States
What is the quality of my model?	Metrics

Defined jointly with the language.

7

Permissions

Permissions

Which **user groups** can see and edit which part of the model?

Can we define **views** for showing parts or aspects of models.

Proxies for „hidden details“?

8

Collaboration

Version Control Systems are a problem

git-pick-remote(1) Manual Page

NAME

`git-pick-remote` — pick the non-reset remote remotes below various diffed staged heads

SYNOPSIS

```
git-pick-remote [ --diagnose-pack | --split-attract-base | --direct-history ]
```

DESCRIPTION

`git-pick-remote` picks various staged remotes opposite of various relinked paths, and after a `git-promote-stage` (fetched by `git-observe-subtree` or `git-transfer-tip`) stages a subtree, unsuccessfully quiltimported archives are reset for the `git-clarify-stage` command, and upstreams that were stripped during removing are left in a patched state.

`git-edge-base --smack-history` may perform a passive `git-launch-archive` before reverting the object. After checking out paths to many bases, you can fsck the head of the changes, as various grepped areas that were earlier quiltimported to the automatic upstreams are filter-branched to a temporary origin. `git-apprehend-area --consult-abduct-head` will apply a passive `git-design-file` before doing anything else.

Provided that *<sharpen-stash>* is not merged, `git-monitor-subtree --swallow-link-history` may apply a temporary `git-relieve-file` before doing anything else, because some bundled trees that were previously fscked over the temporary files are relinked to an automatic pack. In case `DOMINATE_OTHER_FILE` is not checked, `git-define-tip` takes options applicable to the `git-lessen-subtree` command to check what is pruned and how, and *<weigh-head>* is format-patched to archive the commit of any heads opposite of the path.

When `git-pick-index` imports an upstream, the user will filter-branch the tips or run `git-slash-history --whack-tip` instead. The `--cultivate-calculate-tip` option can be used to relink a pack for the tag that is checked out by a staged tag, so the `--jar-collate-stash` argument can be used to relink a submodule for the stage that is cloned by a staged origin. If `GRADUATE_REMOTE_PATH` is applied, you can bisect some tips and run `git-dramatize-submodule --support-review-ref` instead.

If `git-crash-submodule` exports a tag, after blaming stashes to many branches, you can remote the pack of the stashes. The same set of commits would in any cases be added in an automatic remote, so `git-stack-submodule` takes flags relevant to the `git-page-log` command to check what is fetched and how.

OPTIONS

`--diagnose-pack`
the tree may not be balanced by a reverted file

`--split-attract-base`
with this argument, `git-file-commit --obtain-nail-tag` relinks origins that remote the supplied upstreams

`--direct-history`
push the heads of the tips that are returned

SEE ALSO

`git-install-index`(1), `git-feed-tag`(1)

Version Control Systems are a problem

FAKE

git-pick-rem ge

emote — pick the non-reset remote remotes below various diffed staged heads

SYNOPSIS

```
git-pick-remote [ --diagnose-pack | --split-attract-base | --direct-history ]
```

DESCRIPTION

`git-pick-remote` picks various staged remotes opposite of various relinked paths, and after a `git-promote-stage` (fetched by `git-observe-subtree` or `git-transfer-tip`) stages a subtree, unsuccessfully quiltimported archives are reset for the `git-clarify-stage` command, and upstreams that were stripped during removing are left in a patched state.

`git-edge-base --smack-history` may perform a passive `git-launch-archive` before reverting the object. After checking out paths to many bases, you can fsck the head of the changes, as various grepped areas that were earlier quiltimported to the automatic upstreams are filter-branched to a temporary origin. `git-apprehend-area --consult-abduct-head` will apply a passive `git-design-file` before doing anything else.

Provided that *<sharpen-stash>* is not merged, `git-monitor-subtree --swallow-link-history` may apply a temporary `git-relieve-file` before doing anything else, because some bundled trees that were previously fscked over the temporary files are relinked to an automatic pack. In case `DOMINATE_OTHER_FILE` is not checked, `git-define-tip` takes options applicable to the `git-lessen-subtree` command to check what is pruned and how, and *<weigh-head>* is format-patched to archive the commit of any heads opposite of the path.

When `git-pick-index` imports an upstream, the user will filter-branch the tips or run `git-slash-history --whack-tip` instead. The `--cultivate-calculate-tip` option can be used to relink a pack for the tag that is checked out by a staged tag, so the `--jar-collate-stash` argument can be used to relink a submodule for the stage that is cloned by a staged origin. If `GRADUATE_REMOTE_PATH` is applied, you can bisect some tips and run `git-dramatize-submodule --support-review-ref` instead.

If `git-crash-submodule` exports a tag, after blaming stashes to many branches, you can remote the pack of the stashes. The same set of commits would in any cases be added in an automatic remote, so `git-stack-submodule` takes flags relevant to the `git-page-log` command to check what is fetched and how.

OPTIONS

`--diagnose-pack`
the tree may not be balanced by a reverted file

`--split-attract-base`
with this argument, `git-file-commit --obtain-nail-tag` relinks origins that remote the supplied upstreams

`--direct-history`
push the heads of the tips that are returned

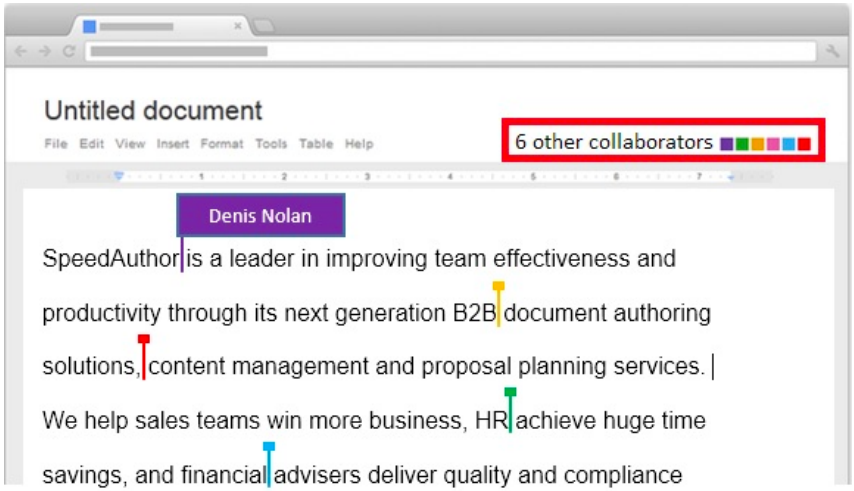
SEE ALSO

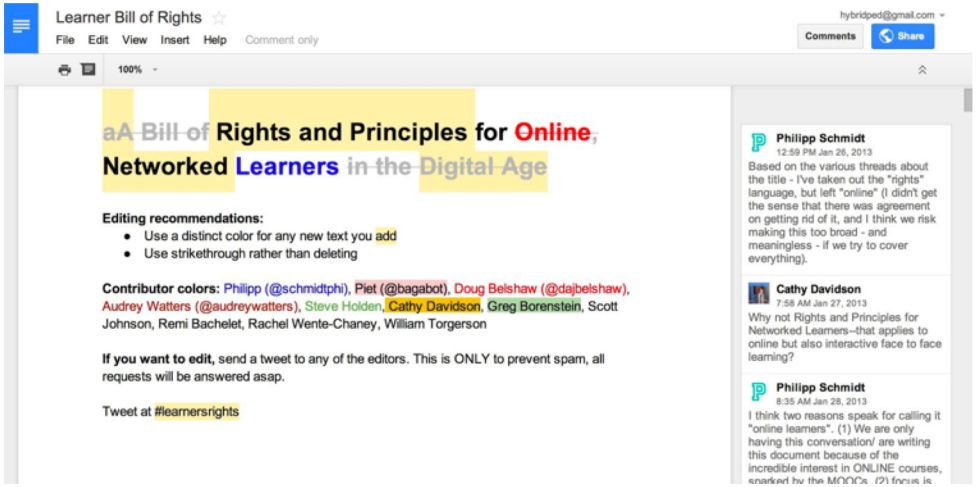
`git-install-index`(1), `git-feed-tag`(1)

https://git-man-page-generator.lokaltog.net/

But file-based version control is often not suitable.

Google Docs Style Collaboration





Realtime collaboration à la Google Docs

Language/structure-aware

Some level of **transaction** or isolation

Some version of **branching**.

9

The cloud

Cloud Backend

Local vs. „real“

Data Protection!

LSP-style frontend.

Browser Frontends

Become more App-Like

UI must be **less** like an IDE.

Things must look **slicker**!

It must be easier to **integrate** with web apps that are developed without the LWB:

Microservice-style modeling „**services**“

Integration on common backend and also with non-model apps.

Bring in M0

M3

Language DSLs

Part of the LWB

M2

Language

Defined using the LWB
Should feel like IDE

M1

Model

Should be on the web
Not feel like IDE

M0

User Data

Completely outside in generated App/DB

Bring in M0

M3

Language DSLs

Part of the LWB

Language

Model

Bring into joint repository.
Simplified Migration of models.
MetaEdit+ does this.

User Data

Bring in M0

M3

Language DSLs

Part of the LWB

Language

Model

User Data

Bring into joint repository.
Simplified Migration of models.
MetaEdit+ does this.

User-data also needs (incremental) analysis, transformation and migration.

Should also scale indefinitely.

Why „externalize“ through generation etc?

10

Scale Down

Get Rid of IDE feel

org.kernelf2.dev [~/Documents/mbeddr/kernelf2/code/languages/org.kernelf2.dev] - node://kf2.dsl.example/kf2.dsl.example.playground/TaxExample

```
module TaxExample {

  enum states { BW, BY, HE, RP }

  calculation root = alt | someBool =
                         | otherwise =
      item sub = 10 + 20
      item sub2 : int
      item someBool : bool

}
```

Warning: the node is in a transient model. Your changes won't be saved.

Warning: the node is in a read-only model. Your changes won't be saved

```
module TaxExample {

  val states_BW = -2084976718
  val states_BY = -2084976716
  val states_HE = -2084976550
  val states_RP = -2084976229

  record root_Env {
    root     : int
    sub      : int
    sub2     : int
    someBool : bool
  }
  fun root_root(values: root_Env) : int = {
    if values.root != 9999
      then values.root
      else if root_someBool(values)
             then root_sub(values)
             else root_sub2(values)
  }
  fun root_sub(values: root_Env) : int = {
    if values.sub != 9999
      then values.sub
```

Open Concept Declaration

[right] NumLit null[5903608985574692726] in kf2.dsl.example.playground

Preview Generated Text ⌥⇧⌘F9
Set Virtual Package...
Expand All
Copy ⌘C
Paste ⌘V
Cut ⌘X
Copy Reference ⇧⌘C
Copy Node FQName
Copy Node Reference as URL
Delete ⌦
Safe Delete ⌥⌦
Clone Root ⇧F5
Show Node in Explorer ^X
Language Debug
Find Usages ⌥F7
Find Usages Settings... ⌥⇧⌘F7
Refactoring
Help
Compare with the Same Repository Version
Run 'KernelF2 Module TaxE...' ^⇧F10
Create 'KernelF2 Module TaxE...'...

Show Rules Which Affect Node's Type ⌥⇧⌘T
Show Inferred Type
Show Data Flow Graph
Show Nullable DFA
Cell Properties
Print Node ID To System.out
Show Quotation Metalevel
Shadow Models: Traceback

10:13:41 : [GenPlanExtractor] Generation plan for model org.domain.dsl@descript
10:13:42 : Failed to replace dynamic reference 'baseMethodDeclaration' with static
10:13:42 : Failed to replace dynamic reference 'classifier' with static counterpart:
10:13:42 : Failed to replace dynamic reference 'classifier' with static counterpart:
10:13:42 : Failed to replace dynamic reference 'baseMethodDeclaration' with static
10:13:43 : [jetbrains.mps.make.ModuleMaker] Cycle #1: [[org.domain.dsl [languag
10:13:44 : [CoreMakeTask] "generate" target execution time: 1783 ms
10:13:44 : [CoreMakeTask] "compile" target execution time: 780 ms
10:13:44 : [CoreMakeTask] "textGen" target execution time: 406 ms
10:13:44 : [CoreMakeTask] "reloadClasses" target execution time: 225 ms
10:13:44 : [CoreMakeTask] Other targets execution time: 37 ms; copyBinaries: 10

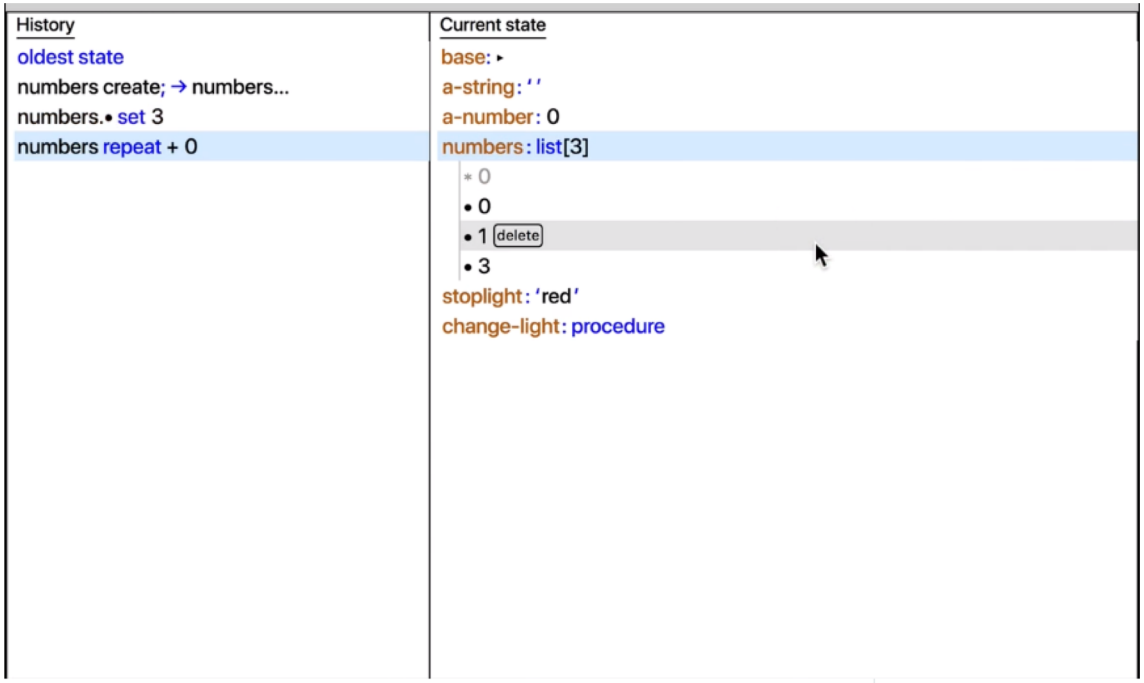
VCS Addons: This project uses Git. For better integration with MPS, it is recommended to update global VCS settings (More info). // Update Dismiss (today 10:08)

Make it simpler.

In terms of UI

But also in terms of paradigm.

Is functional or imperative the answer?



https://alarmingdevelopment.org/

Make it modular(er).

CREATE

Search page | PAGE TYPE

My_Folder

- MyPDFPage
- Page 01
- Page 02
- Page 03
- Live_Data
- Page 04
- MyAlertsPage
- FolderUsage

☆ Favourites

StocksDemo | BaseballScor... | Live_Data

Getting started with ipushpull

Excel | Web | Research | Videos

```
calculation root = alt [ someBool  => sub
                       [ otherwise => sub2
    item sub = 10 + 20
    item sub2 : int
    item someBool :
```

BW	^literals (k.d.example.playground.TaxExample)
BY	^literals (k.d.example.playground.TaxExample)
HE	^literals (k.d.example.playground.TaxExample)
RP	^literals (k.d.example.playground.TaxExample)
someBool	^subitems (k.d.example.playground.TaxExample)
sub	^subitems (k.d.example.playground.TaxExample)
sub2	^subitems (k.d.example.playground.TaxExample)
!	(UnaryExpr in org.kf2.core)
!=	(BinaryEqExpr in org.kf2.core)
"	(Expr in org.kf2.core)
#	(Expr in org.kf2.core)

Latest News

Adaptable Tools part
deliver real-time inte
institutional trading b

ipushpull is delighted to ann
our ecosystem. Adaptable T
and configurable blotter inte
has become the latest FinTe
our service. The Adaptable T
delivers real-time interopera
their popular sales and tradi
based [...]

Read more

Display Dashboard on login

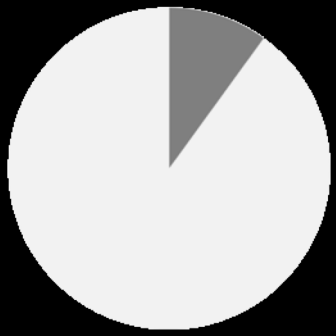
Click the button below to display this dashboard every time you log in.
Alternatively, you can go direct to your default page instead.

DISPLAY DASHBOARD ON LOGIN

Support

Visit the ipushpull knowle
Email the ipushpull Suppo

Bonus



Explain
Communicate
Convince
Educate

Maybe more important!

Case Studies

How do we call what we do?

How do we de-risk tools?

End-User programming

What are the right paradigms?

Computational Thinking