

Designing a Library for Model Analyses using MPS

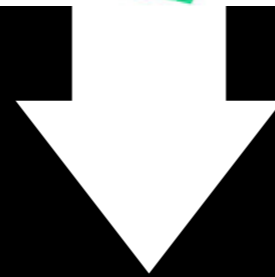
Anthony Anjorin

For a long time there's been a style of software development, **language oriented programming**, that seeks to describe software systems using a collection of **domain specific languages** (e.g., the Unix tradition of “little languages”)

Language Workbenches are a new breed of tools that aim to make language oriented programming a **modern** and **viable** approach

Martin Fowler

(<https://www.martinfowler.com/articles/languageWorkbench.html>)





```
double midnight2(int32 a, int32 b, int32 c) {  
    return 
$$\frac{-b + \sqrt{b^2 - \sum_{i=1}^4 a * c}}{2 * a};$$
  
} midnight2 (function)
```



MPS

—



engineering the future of embedded software

```
double midnight2(int32 a, int32 b, int32 c) {
    return  $\frac{-b + \sqrt{b^2 - \sum_{i=1}^4 a * c}}{2 * a}$ ;
}
midnight2 (function)
```

		Events	
		next(Trackpoint* tp)	reset()
States	beforeFlight	[tp->alt > 0 m] -> airborne	
	airborne	[tp->alt == 0 m && tp->speed == 0 mps] -> crashed	[] -> beforeFlight
		[tp->alt == 0 m && tp->speed > 0 mps] -> landing	
		[tp->speed > 200 mps && tp->alt == 0 m] -> airborne	
		[tp->speed > 100 mps && tp->speed <= 200 mps && tp->alt == 0 m] -> airborne	
landing	[tp->speed == 0 mps] -> landed	[] -> beforeFlight	
	[tp->speed > 0 mps] -> landing		
landed		[] -> beforeFlight	
crashed			



MPS



```
double midnight2(int32 a, int32 b, int32 c) {
    return 
$$\frac{-b + \sqrt{b^2 - \sum_{i=1}^4 a * c}}{2 * a};$$

} midnight2 (function)
```

	Events	reset()
beforeflight	next(trackpoint* tp)	
airborne	[tp->alt > 0 m] -> airborne	
	[tp->alt == 0 m && tp->speed == 0 mps] -> crashed	[] -> beforeflight
	[tp->alt == 0 m && tp->speed > 0 mps] -> landing	
	[tp->speed > 200 mps && tp->alt == 0 m] -> airborne	
States	[tp->speed > 100 mps && tp->speed <= 200 mps && tp->alt == 0 m] -> airborne	
landing	[tp->speed == 0 mps] -> landed	[] -> beforeflight
landed	[tp->speed > 0 mps] -> landing	[] -> beforeflight
crashed		[] -> beforeflight

1 | Once a flight lifts off, you get 100 points

`PointsForTakeoff /functional: tags`

[... points are multiplied by the `$req(PointsFactor)`, discussed below.]

2.1 Hello, World

This tutorial showcases many of the features of mbeddr in an integrated example. The sources ZIP `com.mbeddr.tutorial.zip` is available from the download page at `mbeddr.com`. It is also part of the complete distro package.

Here is a comment.

23/06/14 15:58 (2 min ago) by markusvoelter

And a reply to it.

23/06/14 15:58 (2 min ago) by markusvoelter



MPS



engineering the future of embedded software

```
double midnight2(int32 a, int32 b, int32 c) {
    return 
$$\frac{-b + \sqrt{b^2 - \sum_{i=1}^4 a * c}}{2 * a};$$

} midnight2 (function)
```

	Events	reset()
beforeFlight	next(trackpoint* tp)	
airborne	[tp->alt > 0] -> airborne	
	[tp->alt == 0 && tp->speed == 0 mps] -> crashed	[] -> beforeFlight
	[tp->alt == 0 && tp->speed > 0 mps] -> landing	
	[tp->speed > 200 mps && tp->alt == 0 m] -> airborne	
	[tp->speed > 100 mps && tp->speed <= 200 mps && tp->alt == 0 m] -> airborne	
States		
landing	tp->alt == 0 m] -> airborne	[] -> beforeFlight
	[tp->speed == 0 mps] -> landed	
landed	[tp->speed > 0 mps] -> landing	[] -> beforeFlight
crashed		[] -> beforeFlight

1 | Once a flight lifts off, you get 100 points
 PointsForTakeoff /functional: tags
 [...] points are multiplied by the \$req(PointsFactor), discussed below.]

2.1 Hello, World

This tutorial showcases many of the features of mbeddr in an integrated example. The sources ZIP [com.mbeddr.tutorial.zip](#) is available from the download page at [mbeddr.com](#). It is also part of the complete distro package.

Here is a comment.
 21/08/24 15:58 (2 min ago) by markusvoelter

And a reply to it.
 21/08/24 15:58 (2 min ago) by markusvoelter

```
compositeblock Experiment extends <no extends>
  [double a] => [double result]
  [double b]
  parameters [double limit;]
```

The diagram shows a block diagram for the 'Experiment' composite block. It has three input ports: 'a', 'b', and 'c'. 'a' and 'b' are connected to an 'Adder' block. The output of the 'Adder' block is connected to a 'Max' block. The 'c' input is also connected to the 'Max' block. The output of the 'Max' block is labeled 'result'.



MPS



engineering the future of embedded software

```
double midnight2(int32 a, int32 b, int32 c) {
    return 
$$\frac{-b + \sqrt{b^2 - \sum_{i=1}^4 a * c}}{2 * a};$$

} midnight2 (function)
```

	Events	reset()
beforeFlight	next(trackpoint* tp)	
airborne	[tp->alt > 0 m] -> airborne	
	[tp->alt == 0 m && tp->speed == 0 mps] -> crashed	[] -> beforeFlight
	[tp->alt == 0 m && tp->speed > 0 mps] -> landing	
	[tp->speed > 200 mps && tp->alt == 0 m] -> airborne	
	[tp->speed > 100 mps && tp->speed <= 200 mps && tp->alt == 0 m] -> airborne	
landing	tp->alt == 0 m] -> airborne	[] -> beforeFlight
	[tp->speed == 0 mps] -> landed	
	[tp->speed > 0 mps] -> landing	
landed		[] -> beforeFlight
crashed		[] -> beforeFlight

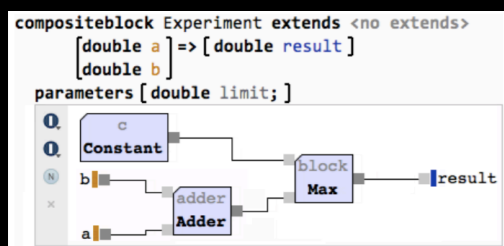
1 | Once a flight lifts off, you get 100 points
 PointsForTakeoff /functional: tags
 [...] points are multiplied by the \$freq(PointsFactor), discussed below.]

2.1 Hello, World

This tutorial showcases many of the features of mbeddr in an integrated example. The sources ZIP [com.mbeddr.tutorial.zip](#) is available from the download page at [mbeddr.com](#). It is also part of the complete distro package.

Here is a comment.
 21/09/14 15:58 (2 min ago) by markusvoelter

And a reply to it.
 21/09/14 15:58 (2 min ago) by markusvoelter



sorted: must be ok: hide ok ones:

last updated: May 7, 2014 (4 months ago) by markusvoelter

FlightJudgementRules

Fastertan100.impl (1)	24	<div style="width: 24%; background-color: blue;"></div>
Fastertan200.impl (1)	32	<div style="width: 32%; background-color: orange;"></div>
InitialNoPoints.inital (1)	8	<div style="width: 8%;"></div>



MPS



```
double midnight2(int32 a, int32 b, int32 c) {
    return 
$$\frac{-b + \sqrt{b^2 - \sum_{i=1}^4 a * c}}{2 * a};$$

} midnight2 (function)
```

	Events	reset()
beforeFlight	[tp->alt > 0] -> airborne	[] -> beforeFlight
airborne	[tp->alt == 0 && tp->speed == 0 mps] -> crashed [tp->alt == 0 && tp->speed > 0 mps] -> landing [tp->speed > 200 mps && tp->alt == 0 m] -> airborne [tp->speed > 100 mps && tp->speed <= 200 mps && tp->alt == 0 m] -> airborne	[] -> beforeFlight
landing	[tp->speed == 0 mps] -> landed [tp->speed > 0 mps] -> landing	[] -> beforeFlight
landed		[] -> beforeFlight
crashed		[] -> beforeFlight

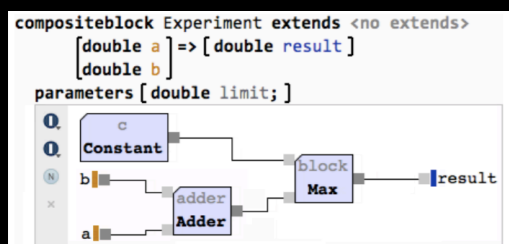
1 | Once a flight lifts off, you get 100 points
PointsForTakeoff /functional: tags
[... points are multiplied by the \$req(PointsFactor), discussed below.]

2.1 Hello, World

This tutorial showcases many of the features of mbeddr in an integrated example. The sources ZIP [com.mbeddr.tutorial.zip](#) is available from the download page at [mbeddr.com](#). It is also part of the complete distro package.

Here is a comment.
21/06/14 15:58 (2 min ago) by markusvoelter

And a reply to it.
21/06/14 15:58 (2 min ago) by markusvoelter



sorted: must be ok: hide ok ones:

last updated: May 7, 2014 (4 months ago) by markusvoelter

FlightJudgementRules	Count	Progress
FasterThan100.impl (1)	24	<div style="width: 100%;"></div>
FasterThan200.impl (1)	32	<div style="width: 100%;"></div>
InitialNoPoints.initial (1)	8	<div style="width: 100%;"></div>

atomicblock Adder realizes IAdder

$$\left[\begin{array}{c} \text{double } a \\ \text{double } b \end{array} \right] \Rightarrow \left[\text{double } res \right]$$

contract $\left[\begin{array}{l} \text{pre}(0) \text{ positive_a: } a > 0; \\ \text{pre}(1) \text{ positive_b: } b > 0; \\ \hline \text{post}(0) \text{ sum: } res == a + b; \end{array} \right]$

ccode { res = a + b; };



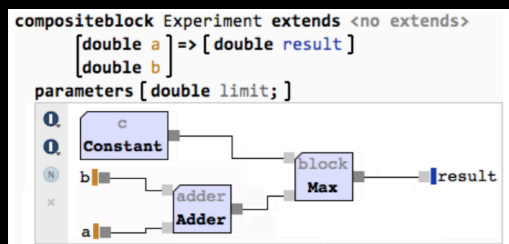
```
double midnight2(int32 a, int32 b, int32 c) {
    return 
$$\frac{-b + \sqrt{b^2 - \sum_{i=1}^4 a * c}}{2 * a};$$

} midnight2 (function)
```

	Events	reset()
beforeFlight	next(trackpoint* tp)	
airborne	[tp->alt > 0] -> airborne [tp->alt == 0 && tp->speed == 0 mps] -> crashed [] -> beforeFlight [tp->alt == 0 && tp->speed > 0 mps] -> landing [tp->speed > 200 mps && tp->alt == 0 m] -> airborne [tp->speed > 100 mps && tp->speed <= 200 mps && tp->alt == 0 m] -> airborne	
landing	[tp->speed == 0 mps] -> landed [tp->speed > 0 mps] -> landing [] -> beforeFlight	
landed		
crashed		[] -> beforeFlight

1 | Once a flight lifts off, you get 100 points
PointsForTakeoff /functional: tags
[... points are multiplied by the \$freq(PointsFactor), discussed below.]

2.1 Hello, World
This tutorial showcases many of the features of mbeddr in an integrated example. The sources ZIP com.mbeddr.tutorial.zip is available from the download page at mbeddr.com. It is also part of the complete distro package.



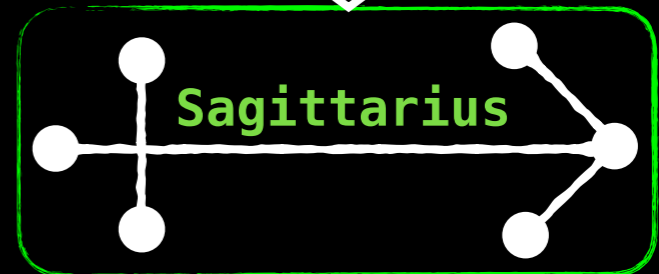
sorted: must be ok: hide ok ones:
last updated: May 7, 2014 (4 months ago) by markusvoelter

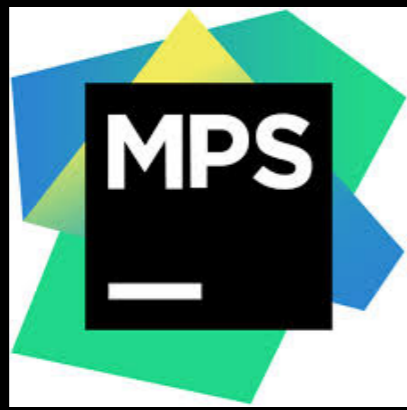
FlightJudgementRules	
FasterThan100.impl (1)	24
FasterThan200.impl (1)	32
InitialNoPoints.initial (1)	8

```
atomicblock Adder realizes IAdder
[double a] => [double res]
[double b]
contract [pre(0) positive_a: a > 0;
pre(1) positive_b: b > 0;
post(0) sum: res == a + b; ]
ccode { res = a + b; };
```

all notations and sub-languages can be **composed** flexibly... even with existing languages such as C, Java, ...

we shall use MPS to build a **model analysis framework** as a seamless extension to Java





1. Get to know a modern and promising **language workbench**
2. Learn how to build, extend, and compose languages in a **model-driven** manner
3. Practice being **creative** in a team