

# DOMENSKO SPECIFIČNI MODELIRNI JEZIKI 2024/2025

1. predavanje

# Učno osebje

- Predavatelj: izr. prof. dr. Tomaž Kosar ([tomaz.kosar@um.si](mailto:tomaz.kosar@um.si))
- Asistent: izr. prof. dr. Tomaž Kosar ([tomaz.kosar@um.si](mailto:tomaz.kosar@um.si))  
NI POMOTA ☺
- Govorilne ure: ponedeljek, 8-10, G3-1N-34 (kabinet 34)
- Tehniški sodelavec:  
Mario Casar, univ.dipl. inž. ([mario.casar@um.si](mailto:mario.casar@um.si))
- Spletna stran predmeta  
<https://estudij.um.si/>

# Obseg predmeta

- Predavanja: 30 ur
  - ponedeljek, 14.00-15.30, DELTA
- Seminarske vaje: 2 uri
  - 4.10.2023, ponedeljek 14:00-15.30, Turing
- Laboratorijske vaje: 28 ur
  - 1. skupina (RV 1): četrtek, 09.30 – 11.00, F-102
  - 2. skupina (RV 2): četrtek, 11.00 – 12.30, F-102
  - 3. skupina (RV 3): petek, 08.00 – 09.30, F-201
  - 4. skupina (RV 4): petek, 09.30 – 11.00, F-201
  - Razdelitev pri asistentu D. Jesenku (Računalniška multimedija)
- Nadomeščanja v oktobru
- Samostojno delo: 90 ur
- ECTS: 5 kreditnih točk

# Način ocenjevanja

## ■ Ocenjevanje

- od 50 do 60 %, ocena 6
- od 60 do 70 %, ocena 7
- od 70 do 80 %, ocena 8
- od 80 do 90 %, ocena 9
- od 90 % , ocena 10

# Način ocenjevanja

## ■ Sprotni način

- Ocena sestavljena iz:
  - opravljene laboratorijske **vaje** 50% (obvezno 25%)
  - **vmesni izpiti** 50% skupno (obvezno 35% pri posameznem)
    - 1. vmesni izpiti: **TODO**
    - 2. vmesni izpiti: **TODO**
  - Prisotnost na predavanjih: 5% (vsaj 11/13)
  - Video po končanih vajah: 2%

## ■ Klasični način

- Ocena sestavljena iz:
  - opravljene laboratorijske **vaje** 50% (obvezno 25%)
  - **pisni izpiti** 50% (obvezno 25%)  
(zimsko, poletno in jesensko izpitno obdobje)

# Način ocenjevanja

- Študent opravi izpit, če zbere skupaj iz vseh postavk najmanj **50 %**.
- Znotraj te vsote mora tako pri **izpitu** kot pri **laboratorijskih vajah** doseči najmanj **polovico** predvidenih obveznosti.

# Vsebina predmeta (1/4)

- Osnove **razvoja** domensko specifičnih (modelirnih) jezikov: analiza domene, načrtovanje domene, implementacija domene.
- **Inženiring domene**: analiza, pristop FODA, načrtovanje (identifikacija komponent, arhitektura domene, konfiguracija komponent, opis konfiguracije z domensko specifičnimi jeziki).

# Vsebina predmeta (2/4)

- **Domensko specifični jeziki:** koncepti domensko specifičnih jezikov, faze razvoja domensko specifičnih jezikov, formalne metode za načrtovanje programskih jezikov, implementacijski vzorci.
- **Orodja** za razvoj domensko specifičnih jezikov (JetBrains MPS).



# Vsebina predmeta (3/4)

- Vizualni jeziki

- Bločni jeziki (angl. block-based languages)

# Vsebina predmeta (4/4)

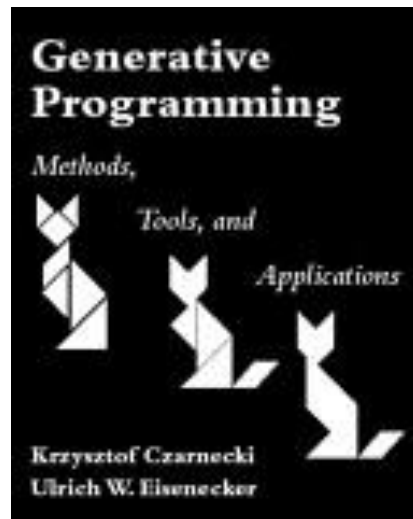
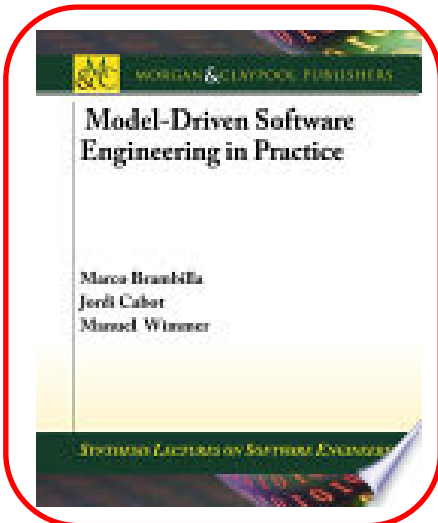
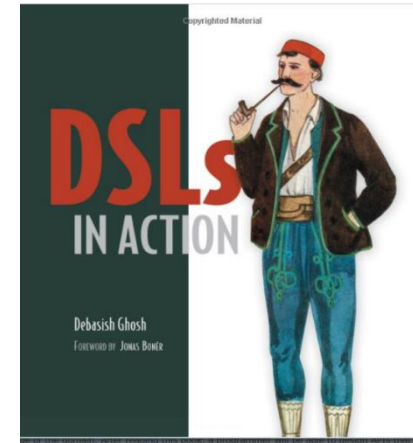
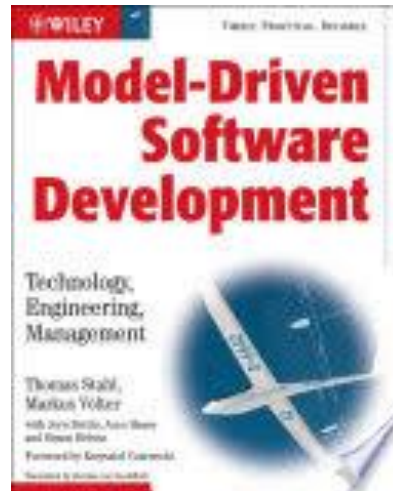
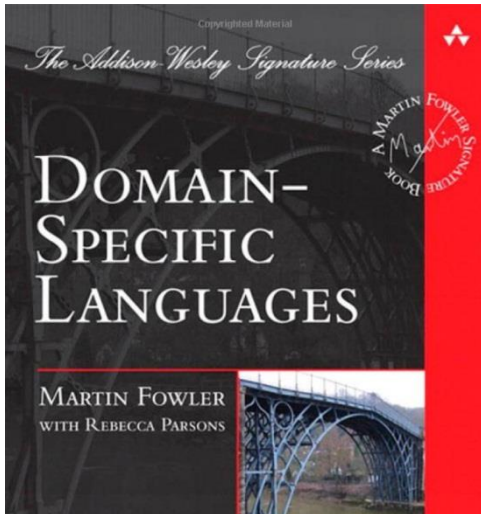
- Domensko specifični **modelirni** jeziki
  - Meta modeliranje
- **Modelno-vodeno inženirstvo**: Razvoj temelječ na modelih.
  - Koncepiranje rešitve.
  - Domenska arhitektura.
  - Razvoj modelirnih jezikov.
- **Sorodne tehnike** (modelno vodeno testiranje, itd.),
- **Orodja** za razvoj domensko specifičnih modelirnih jezikov.

# Temeljni študijski viri

- K. Czarnecki, U. Eisenecker: *Generative Programming: Methods, Tools and Applications*. Addison Wesley, Boston, 2000.
- M. Fowler: *Domain-Specific Languages*, Addison-Wesley, Boston, 2010.
- Thomas Stahl, Markus Voelter, Krzysztof Czarnecki: *Model-Driven Software Development: Technology, Engineering, Management*, John Wiley & Sons, 2006.
- **Marco Brambilla, Jordi Cabot, Manuel Wimmer: Model Driven Software Engineering in Practice, Morgan & Claypool, USA, 2012.**



# Temeljni študijski viri



# Raziskovanje

- Doktorska naloga (2007)
  - Razvoj domensko specifičnih jezikov iz aplikacijskih vmesnikov
- SICRIS ([link](#))
- Google Scholar ([link](#))
- Receptent v revijah (2024):
  - Journal of computer languages (Elsevier)
  - Sensors, Applied Sciences (MDPI)
  - Empirical software Engineering (Springer), itd.
- Programski odbor konferenc:
  - International workshop on human factors in modeling / modeling on human factors (HuFaMo'21, [link](#))
  - International Conference on Program Comprehension ([link](#))
  - FEDCSIS: Workshop on Model Driven Approaches in System Development ([link](#)), Workshop on Advances in Programming Languages
  - MODELSWARD 2024, Model-Based Software and Systems Engineering ([link](#)) , itd.
- Mentorstvo pri magistrskih nalogah:
  - ČOLIĆ, Nikolaj. Sikvel: generator ogrinja podatkovne baze : magistrsko delo : magistrsko delo. (2023, [link](#))
  - FERŠ, Vito. Jezik Blockly za programiranje krmilnikov PLC: magistrsko delo (2019)
- Mentorstvo pri doktorskih nalogah: 2 v teku

# Cilji predmeta

- Cilj tega predmeta je **analizirati metode in tehnike gradnje aplikacij** iz področij:
  - Generativnih metod
  - Domensko specifičnih jezikov
  - Domensko specifičnih modelirnih jezikov

# Predvideni študijski rezultati

- Po zaključku tega predmeta bo študent sposoben
  - načrtovati aplikacije primerne za **družino problemov**,
  - analizirati **aplikacijsko domeno** in poiskati skupne in variabilne dele,
  - **načrtovati in implementirati** domensko-specifične (modelirne) jezike.



# Primer DSLjev

■ Ionic framework: <https://ionicframework.com/>

```
<ion-tabs class="tabs-icon-top">[SEP]
  <!-- My Tab -->
  <ion-tab title="My Expense" icon="icon ion-log-in" href="#/tab/dash">[SEP]
    <ion-nav-view name="tab-dash"></ion-nav-view>[SEP]
  </ion-tab>[SEP]
  <!-- Friends Tab -->
  <ion-tab title="Roomie's" icon="icon ion-log-out" href="#/tab/friends">
    <ion-nav-view name="tab-friends"></ion-nav-view>[SEP]
  </ion-tab>[SEP]
  <!-- Account -->
  <ion-tab title="Account" icon="icon ion-ios7-gear" href="#/tab/account">[SEP]
    <ion-nav-view name="tab-account"></ion-nav-view>[SEP]
  </ion-tab>[SEP]
</ion-tabs>
```

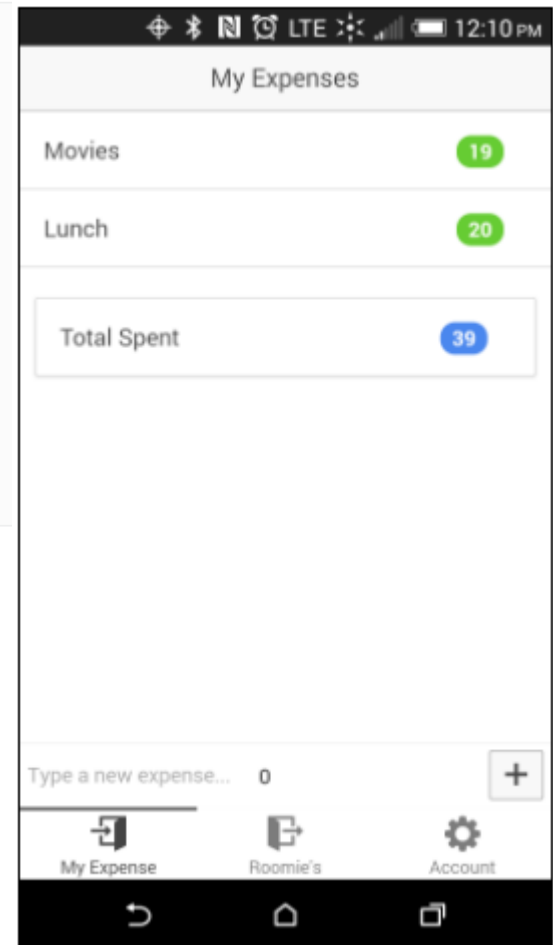




# Primer DSLjev

■ Ionic framework: <https://ionicframework.com/>

```
<ion-tabs class="tabs-icon-top">
  <!-- My Tab -->
  <ion-tab title="My Expense" icon="icon ion-log-in" href="#/tab/dash">
    <ion-nav-view name="tab-dash"></ion-nav-view>
  </ion-tab>
  <!-- Friends Tab -->
  <ion-tab title="Roomie's" icon="icon ion-log-out" href="#/tab/friends">
    <ion-nav-view name="tab-friends"></ion-nav-view>
  </ion-tab>
  <!-- Account -->
  <ion-tab title="Account" icon="icon ion-ios7-gear" href="#/tab/account">
    <ion-nav-view name="tab-account"></ion-nav-view>
  </ion-tab>
</ion-tabs>
```





# Primer DSLjev

■ Markdown in YAML: <https://jekyllrb.com/>

```
2021-10-17-Accommodation.md X
bioma2022 > _posts > 2021-10-17-Accommodation.md > ...
16
17 ## Accommodation
18 To help you decide where to stay, we recommend the following nearby hotels.
19
20 ### S Hotel
21 It is located in the city center, closest to the venue, within walking distance of 100m.
22 Reservations via [e-mail](mailto:info@shotel.si) or [website](http://www.en.shotel.si/).
23
24 ### Hotel Maribor
25 It is located in the city center, within walking distance of 500m from the venue.
26 Reservations via [e-mail](mailto:info@hotelmaribor.si) or [website](https://www.hotelmaribor.si/en/).
27
28 ### Hotel City Maribor
29 It is located in the city center, within walking distance of the venue, 1000m.
30 Reservations via [e-mail](mailto:info@hotelcitymb.si) or [website](https://www.hotelcitymb.si/en/).
31
32 ### Hotel Orel
33 Located in the city center, within walking distance of the venue, 1000m.
34 Reservations via [e-mail](mailto:orel@termemb.si) or the [website](https://www.termemb.si/en/hotels/hotel-orel).
35
36 ### Other hotels
37 There are also other hotels in the vicinity of the venue. You can find more alternatives on the [website](https://www.visitmaribor.si/en/accommodation-and-offer/hotels/).
38
```

```
! _config.yml X
bioma2022 > ! _config.yml
1 # Site Settings
2 name: BIOMA 2022
3 title: BIOMA 2022 | All info about conference
4 description: TThe 10th International Conference on
5 tags:
6   - dates
7   - info
8   - welcome
9   - theme
10 show_hero: true
11 menu:
12   - title: Home
13     url: /
14   - title: About BIOMA Conference
15     url: /AboutConference/
16   - title: Conference Venue
17     url: /ConferenceVenue/
18   - title: Program
19     url: /Program/
20   - title: Photos
21     url: /Photos/
22   - title: Keynote Speakers
23     url: /Speakers/
24   - title: Registration
25     url: /Registration/
26   - title: Accommodation
27     url: /Accommodation/
28   - title: Important Dates
29     url: /ImportantDates/
30   - title: Committees
31     url: /Committees/
32   - title: Author Guidelines
```



# Primer DSLjev

■ Markdown in YAML: <https://jekyllrb.com/>

bioma2022.um.si/Accommodation/

## Accommodation

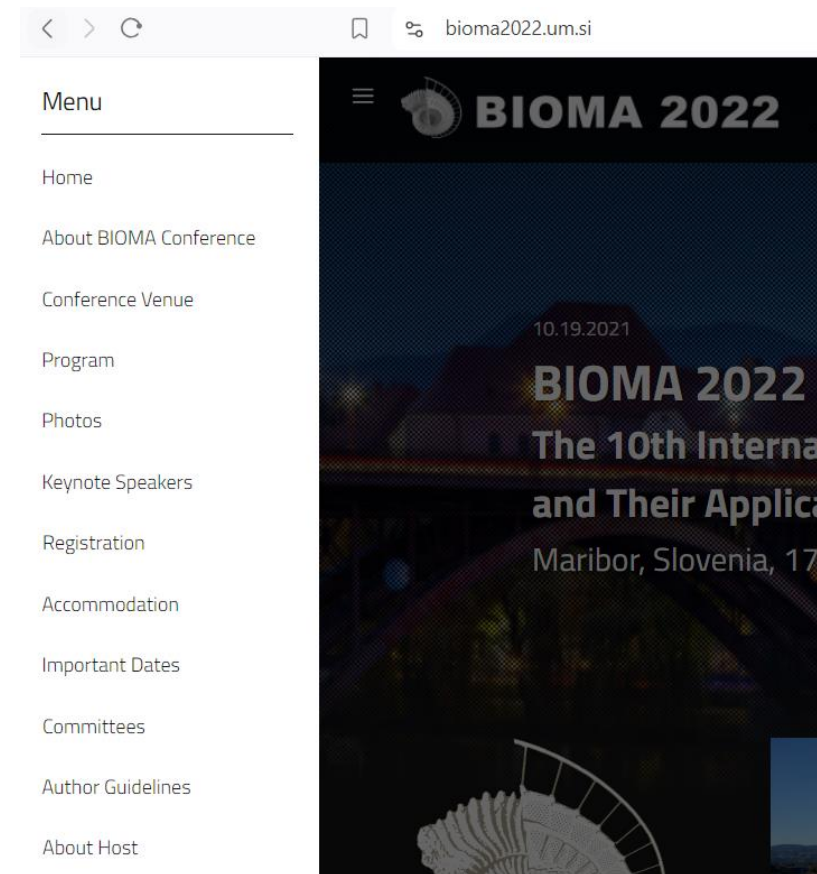
To help you decide where to stay, we recommend the following nearby hotels.

### S Hotel

It is located in the city center, closest to the venue, within walking distance of 100m. Reservations via [e-mail](#) or [website](#).

### Hotel Maribor

It is located in the city center, within walking distance of 500m from the venue. Reservations via [e-mail](#) or [website](#).





# Primer DSLjev

## ■ XAML

```
<Window x:Class = "XAMLVsCode.MainWindow"
  xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x = "http://schemas.microsoft.com/winfx/2006/xaml" Title = "MainWindow" Height =

  <StackPanel>
    <TextBlock Text = "Welcome to XAML Tutorial" Height = "20" Width = "200" Margin = "5" />
    <Button Content = "Ok" Height = "20" Width = "60" Margin = "5"/>
  </StackPanel>

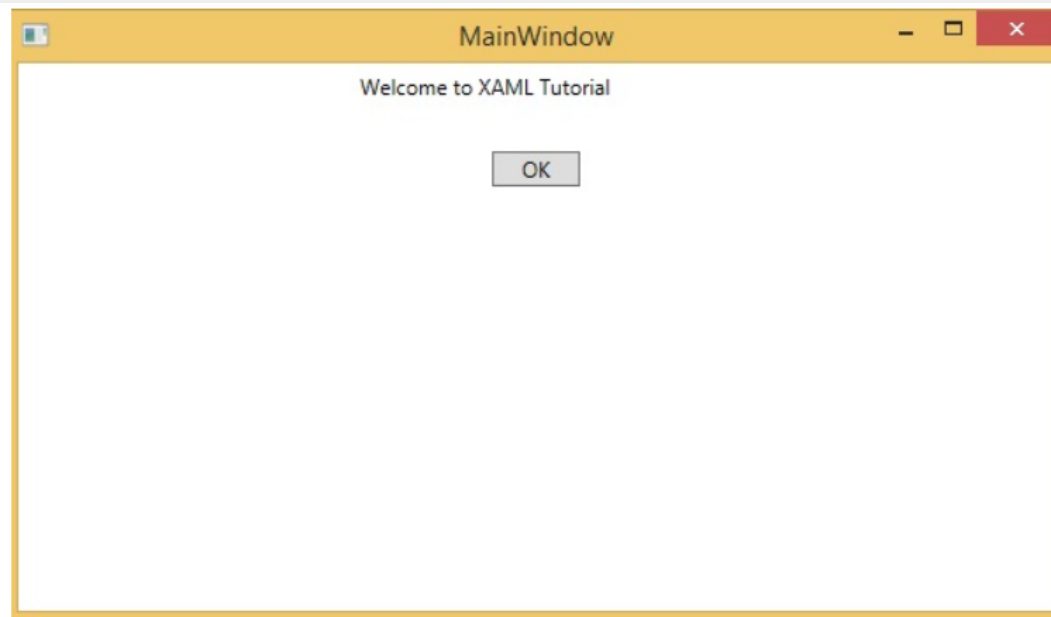
</Window>
```

# Primer DSLjev

## ■ XAML

```
<Window x:Class = "XAMLVsCode.MainWindow"
  xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x = "http://schemas.microsoft.com/winfx/2006/xaml" Title = "MainWindow" Height =

  <StackPanel>
    <TextBlock Text = "Welcome to XAML Tutorial" Height = "20" Width = "200" Margin = "5" />
    <Button Content = "Ok" Height = "20" Width = "60" Margin = "5"/>
  </StackPanel>
</Window>
```





# Primer DSLjev

## ■ XAML

```
<Window x:Class = "XAMLVsCode.MainWindow"
        xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x = "http://schemas.microsoft.com/winfx/2006/xaml" Title = "MainWindow" Height =
        <StackPanel>
            <TextBlock Text = "Welcome to XAML Tutorial" Height = "20" Width = "200" Margin = "5"
            <Button Content = "Ok" Height = "20" Width = "60" Margin = "5"/>
        </StackPanel>
</Window>
```



```
using System;
using System.Text;
using System.Windows;
using System.Windows.Controls;

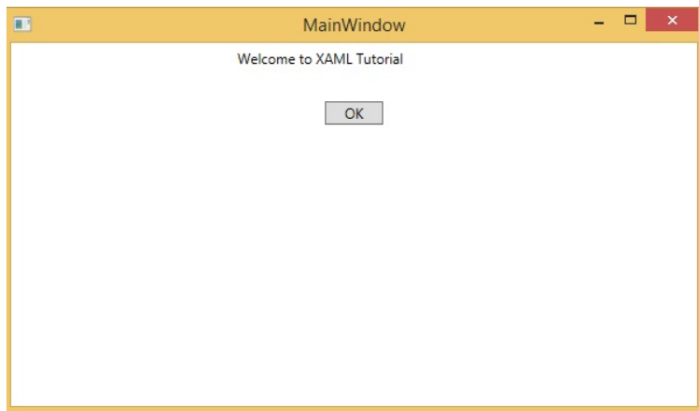
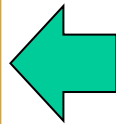
namespace XAMLVsCode {
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>

    public partial class MainWindow : Window {
        public MainWindow() {
            InitializeComponent();

            // Create the StackPanel
            StackPanel stackPanel = new StackPanel();
            this.Content = stackPanel;

            // Create the TextBlock
            TextBlock textBlock = new TextBlock();
            textBlock.Text = "Welcome to XAML Tutorial";
            textBlock.Height = 20;
            textBlock.Width = 200;
            textBlock.Margin = new Thickness(5);
            stackPanel.Children.Add(textBlock);

            // Create the Button
            Button button = new Button();
            button.Content = "OK";
            button.Height = 20;
            button.Width = 50;
            button.Margin = new Thickness(20);
            stackPanel.Children.Add(button);
        }
    }
}
```



# Primer DSLjev

## ■ CLI orodja (npr. angular)

```
ng generate component virtual-machines

hello-angular $ng generate component virtual-machines
CREATE src/app/virtual-machines/virtual-machines.component.html (35 bytes)
CREATE src/app/virtual-machines/virtual-machines.component.spec.ts (692 bytes)
CREATE src/app/virtual-machines/virtual-machines.component.ts (308 bytes)
CREATE src/app/virtual-machines/virtual-machines.component.css (0 bytes)
UPDATE src/app/app.module.ts (550 bytes)
hello-angular $
```



# Primer DSLja

## ■ Povpraševalni jeziki

### ■ Xpath

/bookstore/book[1]/title

/bookstore/book[price>35]/title

```
<?xml version="1.0" encoding="UTF-8"?>

<bookstore>

<book category="cooking">
  <title lang="en">Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
</book>

<book category="children">
  <title lang="en">Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>

<book category="web">
  <title lang="en">XQuery Kick Start</title>
  <author>James McGovern</author>
  <author>Per Bothner</author>
  <author>Kurt Cagle</author>
  <author>James Linn</author>
  <author>Vaidyanathan Nagarajan</author>
  <year>2003</year>
  <price>49.99</price>
</book>

<book category="web">
  <title lang="en">Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
</book>

</bookstore>
```



# Primer DSLjev

## ■ Graphviz:

- <http://www.graphviz.org/>
- <http://www.webgraphviz.com/>

```
digraph "Branch 6." {
    rankdir=TD;
    ordering = "out";

    start          [shape      = point, peripheries = 2          ];
    end             [shape      = point, peripheries = 2          ];
    "1"             [shape      = box   , width = 0.5, height = 0.5];
    "+1"            [shape      = box   , width = 0.5, height = 0.5];
    "+3"            [shape      = box   , width = 0.5, height = 0.5];
    "+5"            [shape      = box   , width = 0.5, height = 0.5];

    start -> "1" [arrowhead = open , arrowtail = none];
    "1"   -> "+1" [arrowhead = open , arrowtail = none];
    "1"   -> "+3" [arrowhead = open , arrowtail = none];
    "+1"  -> "+5" [arrowhead = open , arrowtail = none];
    "+3"  -> end  [arrowhead = open , arrowtail = none];
    "+5"  -> end  [arrowhead = open , arrowtail = none];

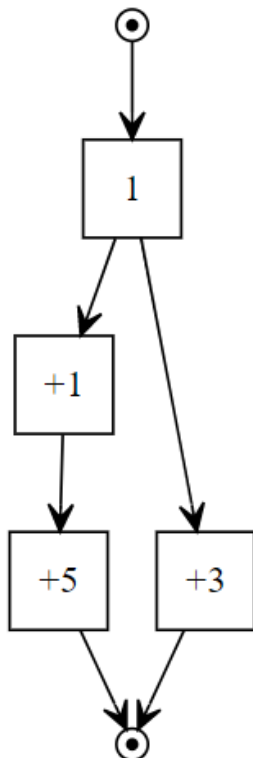
    rank = same; "+1"; "+3";

}
```

# Primer DSLjev

## ■ Graphviz:

- <http://www.graphviz.org/>
- <http://www.webgraphviz.com/>



```

digraph "Branch 6." {
    rankdir=TD;
    ordering = "out";
  
```

```

start      [shape      = point, peripheries = 2           ];
end        [shape      = point, peripheries = 2           ];
"1"        [shape      = box   , width  = 0.5, height = 0.5];
"+1"       [shape      = box   , width  = 0.5, height = 0.5];
"+3"       [shape      = box   , width  = 0.5, height = 0.5];
"+5"       [shape      = box   , width  = 0.5, height = 0.5];
  
```

```

start -> "1" [arrowhead = open , arrowtail = none];
"1"   -> "+1" [arrowhead = open , arrowtail = none];
"1"   -> "+3" [arrowhead = open , arrowtail = none];
"+1"  -> "+5" [arrowhead = open , arrowtail = none];
"+3"  -> end  [arrowhead = open , arrowtail = none];
"+5"  -> end  [arrowhead = open , arrowtail = none];
  
```

```

rank = same; "+5"; "+3";
  
```

```

}
  
```

# DSL EasyWorkFlow (iz vaj)

## ■ Latex – okolje „picture“

```
\begin{picture}(12.000000,23.000000)(-4.750000,-23.000000)
\put(1.2500,-0.5000){\oval(2.5000,1.0000)}
\put(0.0000,-1.0000){\makebox(2.5000,1.0000)[c]{\shortstack[c]{
start
}}}}
\put(1.2500,-1.0000){\vector(0,-1){0.5000}}
\put(-2.7500,-2.5000){\framebox(8.0000,1.0000)[c]{\shortstack[c]{
defining research questions
}}}}
\put(1.2500,-2.5000){\vector(0,-1){0.5000}}
\put(-2.7500,-5.0000){\framebox(8.0000,2.0000)[c]{\shortstack[c]{
defining a search string \&\inclusion/exclusion criteria
}}}}
\put(1.2500,-5.0000){\vector(0,-1){0.5000}}
\put(-1.7500,-7.0000){\line(2,1){3.0000}}
\put(-1.7500,-7.0000){\line(2,-1){3.0000}}
\put(4.2500,-7.0000){\line(-2,-1){3.0000}}
\put(4.2500,-7.0000){\line(-2,1){3.0000}}
\put(-1.7500,-8.5000){\makebox(6.0000,3.0000)[c]{\shortstack[c]{
can more\relevant publications \be found
}}}}
...
```

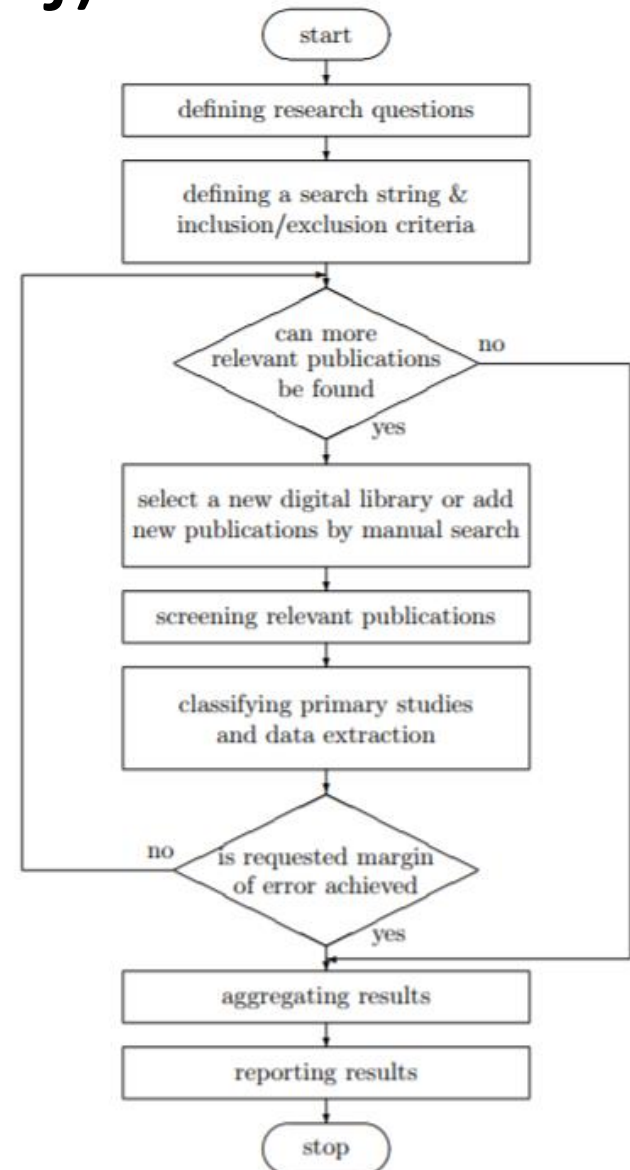
# DSL EasyWorkFlow (iz vaj)

## ■ Latex – okolje „picture“

```

\begin{picture}(12.000000,23.000000)(-4.750000,-23.000000)
\put(1.2500,-0.5000){\oval(2.5000,1.0000)}
\put(0.0000,-1.0000){\makebox(2.5000,1.0000)[c]{\shortstack[c]{
start
}}}
\put(1.2500,-1.0000){\vector(0,-1){0.5000}}
\put(-2.7500,-2.5000){\framebox(8.0000,1.0000)[c]{\shortstack[c]{
defining research questions
}}}
\put(1.2500,-2.5000){\vector(0,-1){0.5000}}
\put(-2.7500,-5.0000){\framebox(8.0000,2.0000)[c]{\shortstack[c]{
defining a search string \&\inclusion/exclusion criteria
}}}
\put(1.2500,-5.0000){\vector(0,-1){0.5000}}
\put(-1.7500,-7.0000){\line(2,1){3.0000}}
\put(-1.7500,-7.0000){\line(2,-1){3.0000}}
\put(4.2500,-7.0000){\line(-2,-1){3.0000}}
\put(4.2500,-7.0000){\line(-2,1){3.0000}}
\put(-1.7500,-8.5000){\makebox(6.0000,3.0000)[c]{\shortstack[c]{
can more\relevant publications \be found
}}}
...

```



# DSL EasyWorkFlow (iz vaj)

## ■ Latex – okolje „picture“

```
test.mydsl
default Skip 1 1 1 1
default Size 2 5

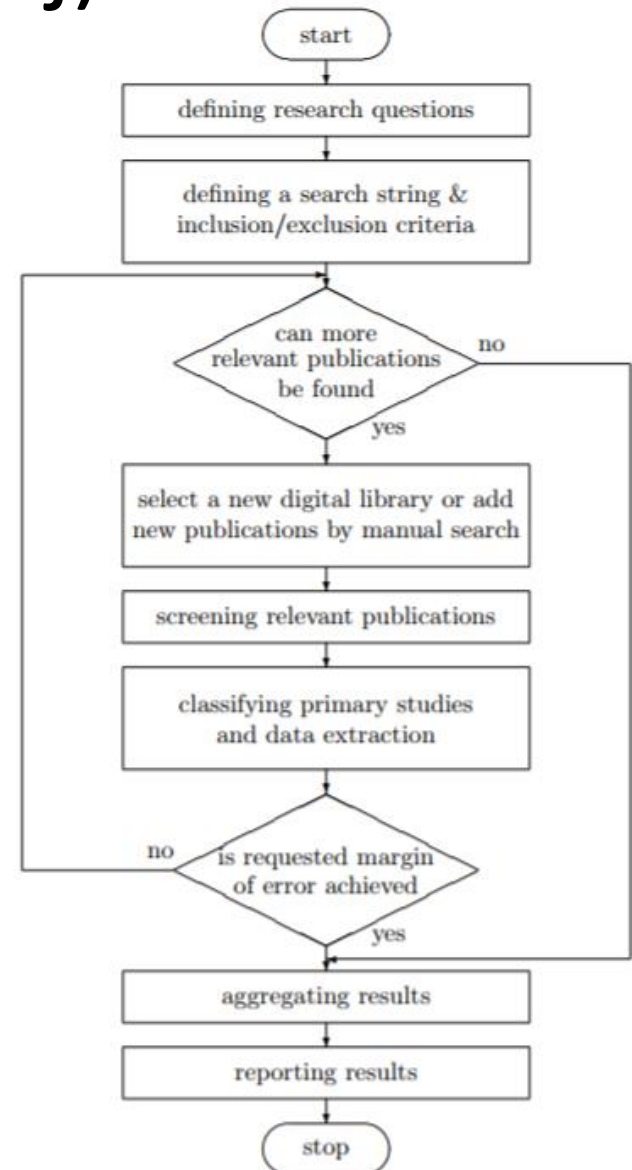
oval "start"
box "defining research questions"
box 4 5 "defining a search string & inclusion/exclusion criteria"

choice . no yes . "can more relevant publications be found"

box 4 5 "select a new digital library or add new publications by manual search"
box "screening relevant publications"
box 4 5 "classifying primary studies and data extraction"

choice . . yes no "is requested margin of error achieved"

box "aggregating results"
box "reporting results "
oval "stop"
```



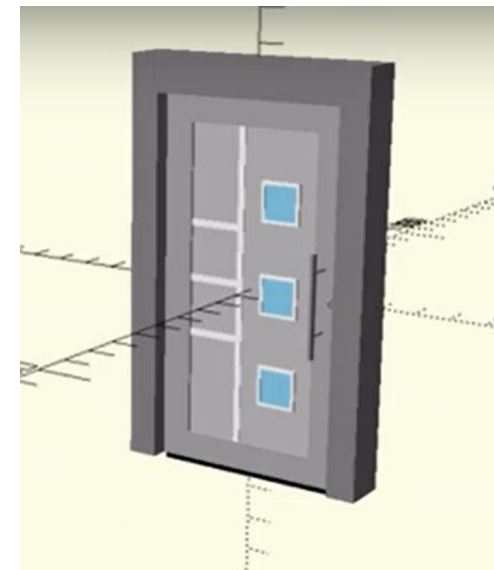
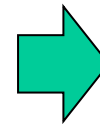
# DSL (3dModeling)

■ Primer: [link](#)

```
door " VrataProgram "  
  shape = 126x220  
  color = metal  
  second_color = shiny metal  
  scale = 1  
  thickness = 1  
  Frame  
    color = silver  
    openness = half  
    thickness = 40  
  Knob  
    type = long  
    color = #000000  
    scale = 1  
    length = 150  
  <no MailSlit>  
  add Roseta  
    color = #000000  
    protrude = 40  
    offset = 20%  
  add Peeper  
    color = #000000  
    offset = 80%  
    protrude = 30  
  add Knocker  
    type = square  
    color = #000000  
    offset = 4%  
  add Filter  
    color = gold  
    number of fins = 6  
    offset = 100%
```



```
include <modules.scad>  
$fa = 1;  
  
door_size = 1.0 * 5;  
door_thick=1.0;  
  
door_color = "metal";  
door_second_color = "metal";  
  
frame_openess = "half";  
frame_thickness=40;  
frame_color= "silver";  
  
knob_type = "long";  
knob_color = "#000000";  
knob_scale = 1.0;  
knob_base_lengths = 150;  
door_base = 220;
```



– OpenSCAD: [link](#)

# Primer vizualnih jezikov

- <https://www.youtube.com/watch?v=ksFfVYclXB4&feature=youtu.be>



# Primer DSLja iz industrijskega projekta

## ■ Cilj

- Celovito **upravljanje varovanja podatkov**
- Optimizacija varovanja (časovno)
- Inteligentno dodajanje naprav
- Možnosti “What-if” scenarijev

## ■ Rešitev

- DSL za opis okolja
- DSL za opis bremen
- Izdelava simulatorja
- Statistika
- Analiza algoritmov
- Samoadaptivnost
- Realno okolje



# Primer DSLja iz industrijskega projekta

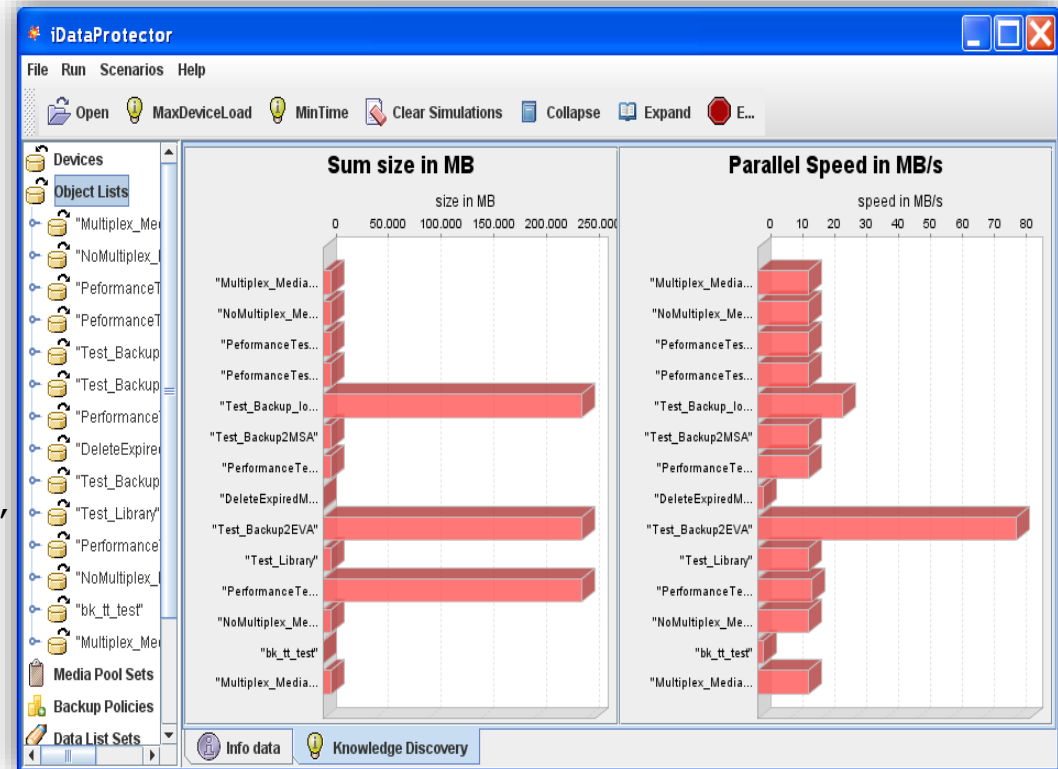
- Opis
  - Grafični
  - Skriptni način dela

```
defMediaTypes (File,DLT,ExaByte,T9940,
               Tape)
defMediaPools (
    "FILE1_Pool" mediaType File,
    "Default T9840" mediaType T9840,
    ... )
backupPolicyDef Policy_ARCHIVE (
    default priority 5 ,
    default poolSet DEFAULT_MP ,
    default startTime 01:30
                    duration 500 min )
setMediaPoolSet NOTEBOOK_MP (
    "DLT1_Pool",
    ...)
```

```
defDevices (
    "DEV" maxSpeed 120 MBs avrSpeed 60 MBs pool "DLT2_Pool" maxParallelObjects 32)

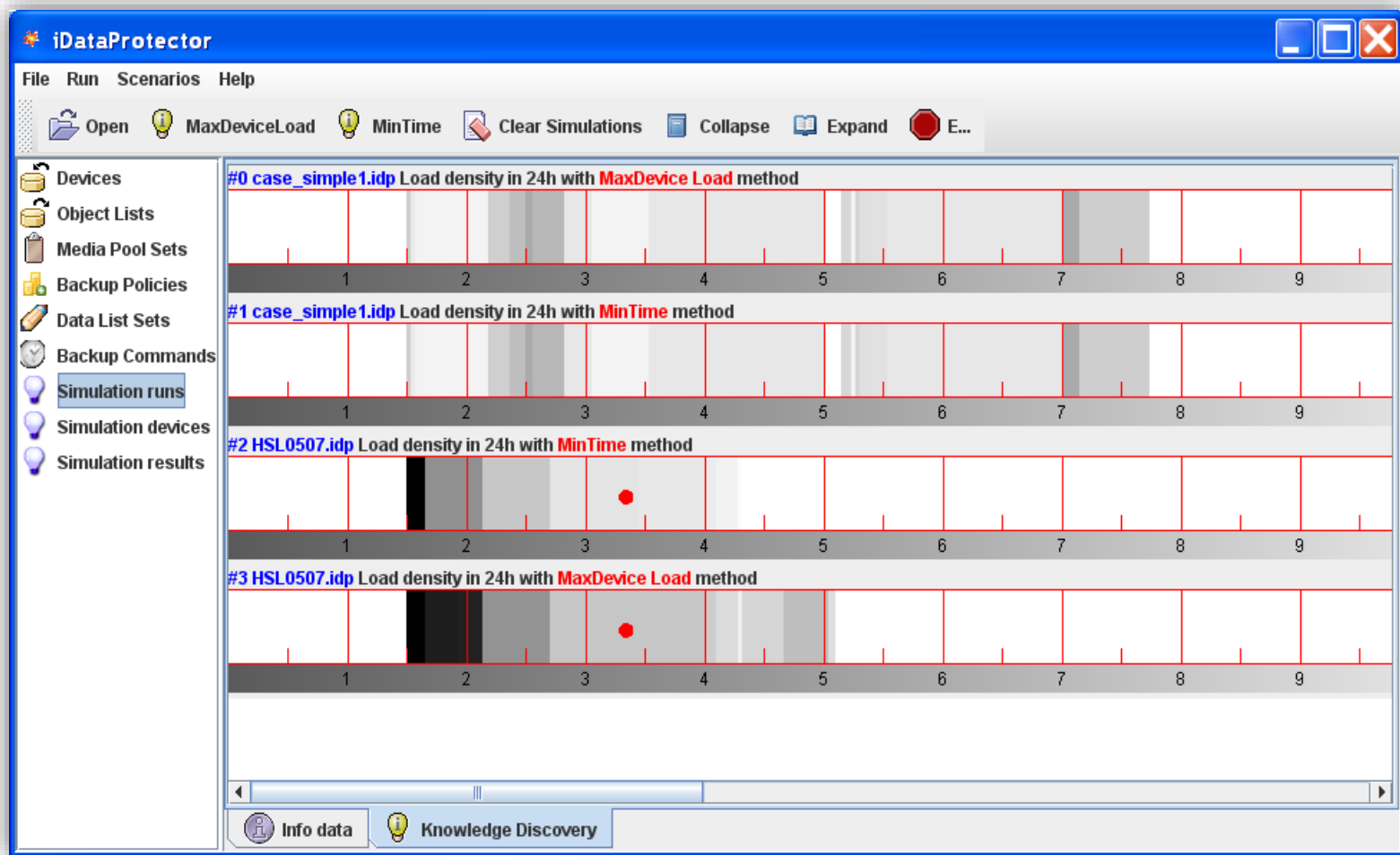
defObjectList name(WINFS "doitName" firma.si:"/C"{"-trees "/c:"}) size 45000 MB,...)

backup dataListSet NOTEBOOK_MP withPolicy Policy_ARCHIVE
```



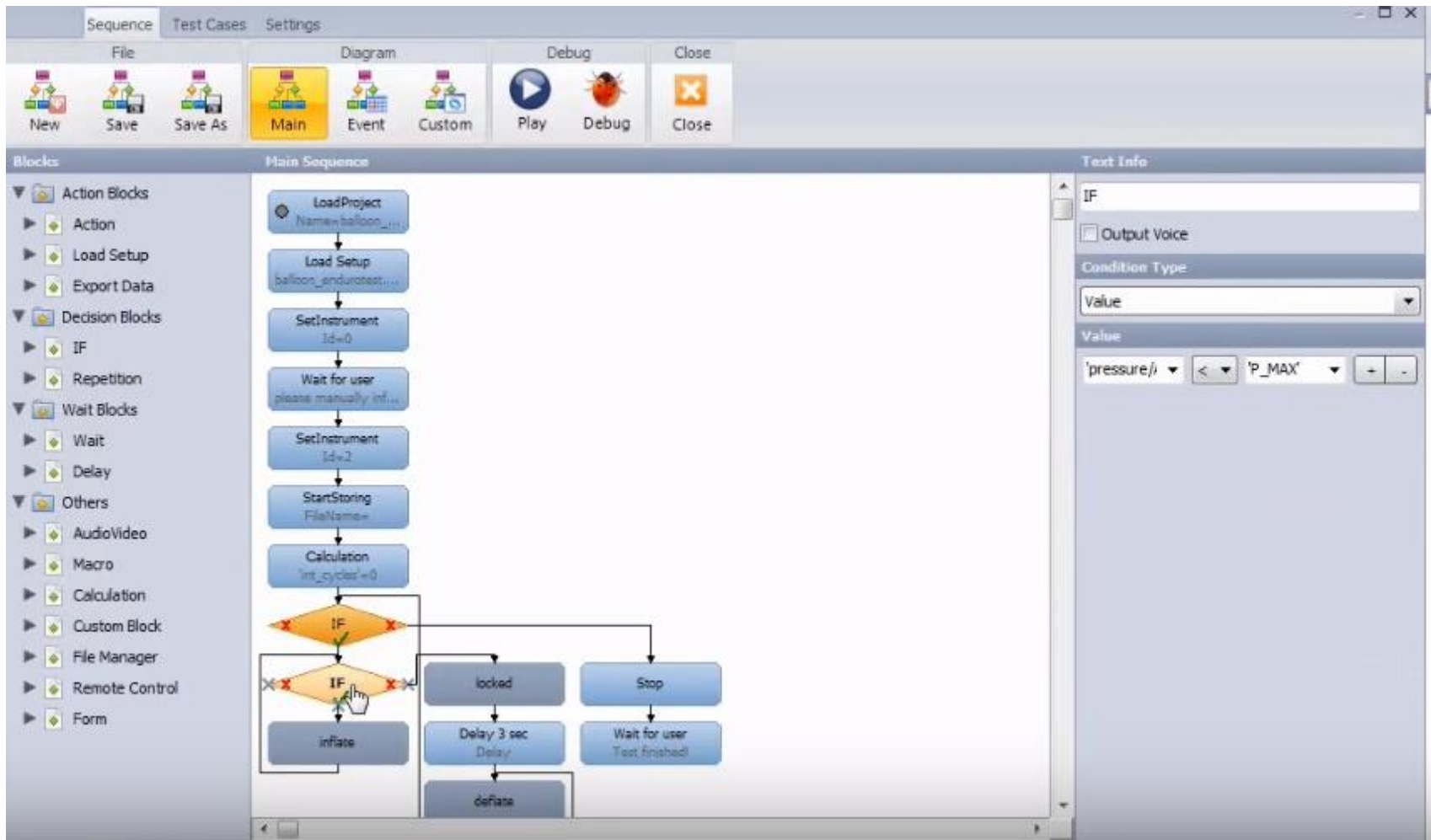
# Primer DSLja iz industrijskega projekta

## ■ Izvajanje aplikacije





# Doktorat iz DSMLjev - Sequencer



# Doktorat iz DSMLjev - Sequencer

## ■ Dewesoft

- doc. dr. Tomaž Kos

## ■ Več o Sequencer-ju

- <https://training.dewesoft.com/online/course/sequencer>
- <https://www.youtube.com/watch?v=ofCSJZu9NxE&t=26s>



# Doktorat iz DSMLjev - Sequencer



# Doktorat iz DSMLjev - Sequencer

- KOS, Tomaž, MERNIK, Marjan, KOSAR, Tomaž. Extending an existing domain-specific modeling language with general-purpose language: example on industrial case-study RT-Sequencer. To be submitted for publication. 2022
- KOS, Tomaž, MERNIK, Marjan, KOSAR, Tomaž. A Tool Support for Model-Driven Development: An Industrial Case Study from a Measurement Domain, vol. 9, iss. 21, 2019, doi: [10.3390/app9214553](https://doi.org/10.3390/app9214553).
- KOS, Tomaž, MERNIK, Marjan, KOSAR, Tomaž. Test automation of a measurement system using a domain-specific modelling language. The Journal of Systems and Software, ISSN 0164-1212. [Print ed.], Jan. 2016, vol. 111, str. 74-88, doi: [10.1016/j.jss.2015.09.002](https://doi.org/10.1016/j.jss.2015.09.002).
- KOSAR, Tomaž, MERNIK, Marjan, GRAY, Jeffrey G., KOS, Tomaž. Debugging measurement systems using a domain-specific modeling language. *Computers in industry*, ISSN 0166-3615. [Print ed.], 2014, vol. 65, iss. 4, str. 622-635, doi: [10.1016/j.compind.2014.01.013](https://doi.org/10.1016/j.compind.2014.01.013).
- KOS, Tomaž, KOSAR, Tomaž, MERNIK, Marjan. Development of data acquisition systems by using a domain-specific modeling language. *Computers in industry*, ISSN 0166-3615. [Print ed.], Apr. 2012, vol. 63, no. 3, str. 181-192, doi: [10.1016/j.compind.2011.09.004](https://doi.org/10.1016/j.compind.2011.09.004)
- KOS, Tomaž, KOSAR, Tomaž, KNEZ, Jure, MERNIK, Marjan. From DCOM interfaces to domain-specific modeling language : a case study on the sequencer. *Computer science and information systems*, ISSN 1820-0214. [Print ed.], May 2011, vol. 8, no. 2, str. 361-378. doi: [10.2298/CSIS101231009K](https://doi.org/10.2298/CSIS101231009K).

# Vaje pri DSML

- Načrtovati majhne jezike
- Lasten
  - DSL
  - Bločni jezik
  - DSML
- Sekundarni plan





# Vaje pri DSML

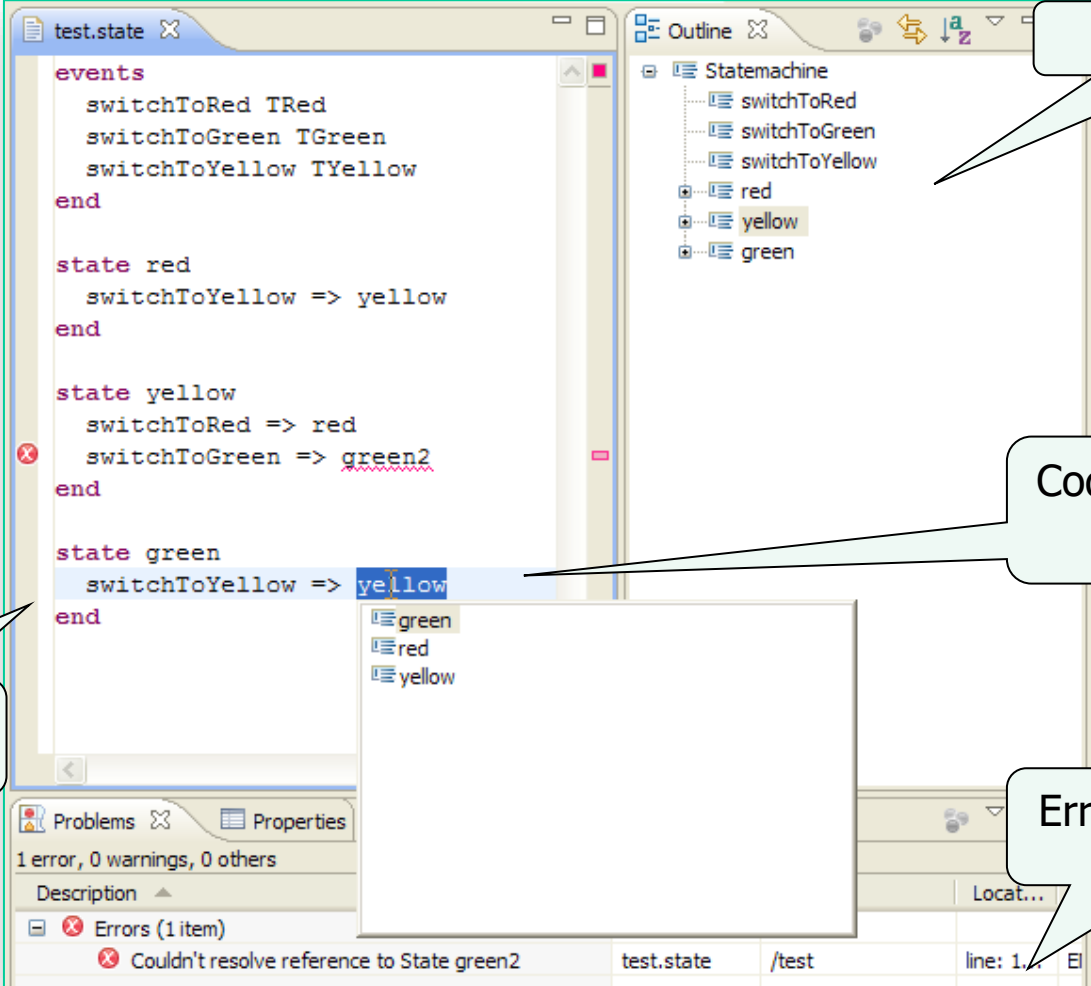
## ■ VIDEO (bonus iz vaj)

The screenshot displays a development environment with three main panes:

- Left Pane (Project Structure):** Shows a project named 'Naloga4' with a sub-project 'KulturaEventsLanguage.sandbox'. The 'KulturaGuide' component is selected.
- Middle Pane (Code Editor):** Displays the code for the 'App\_TextGen' component. The code defines a 'KulturaGuide' component with properties like 'name', 'title', 'description', and 'port'. It also defines a 'components' array with three items: 'Home', 'About', and 'AllEvents', each with its own configuration.
- Right Pane (Browser View):** Shows the running application at 'localhost:3000/home'. The application has a green header with a 'Logout' button. Below the header, there is a main heading 'Welcome to Culture Guide Generated appl' and a paragraph about cultural events. A list of 'Upcoming Events' is displayed, including 'Festival in the Park - Saturday, 15th March', 'Art Exhibition Opening - Friday, 20th March', and 'Live Music Concert - Sunday, 25th March'.



# Vaje pri DSML - anatomija modernih tekstovnih urejevalnikov



The screenshot displays an IDE window titled 'test.state' containing the following code:

```
events
  switchToRed TRed
  switchToGreen TGreen
  switchToYellow TYellow
end

state red
  switchToYellow => yellow
end

state yellow
  switchToRed => red
  switchToGreen => green2
end

state green
  switchToYellow => yellow
end
```

Annotations and features shown:

- Error!**: A red 'x' icon in the left margin indicates an error.
- Highlighting of keywords**: Keywords like 'state', 'end', and 'switchTo' are highlighted in the code.
- Code Completion**: A dropdown menu is visible below the cursor, showing suggestions: 'green', 'red', and 'yellow'.
- Outline View**: A panel on the right shows the project structure, including 'Statemachine' and its components: 'switchToRed', 'switchToGreen', 'switchToYellow', 'red', 'yellow', and 'green'.
- Error Description**: A panel at the bottom shows the error message: 'Couldn't resolve reference to State green2'.

# Uvod

- Izraz **računalniški jezik** združuje veliko različnih jezikov, ki se uporabljajo za **komunikacijo z računalniki**. Je **širši** od bolj pogosto uporabljenega **programskega jezika**.

(Wikipedia)

- Računalniški jeziki
  - Specifikacijski jeziki (formalni jezik za opis sistemov)
  - Modelirni jeziki (grafični in tekstovni, UML)
  - Programski jeziki

# Uvod

## ■ Kaj je programski jezik?

- Jezik za **programiranje računalnikov** (WordNet slovar).
- Programski jezik je **umetni jezik**, ki ga je mogoče uporabiti za nadzor **vedenja stroja**, zlasti računalnika (Wikipedia).
- Programski jezik je zapis za programe.
- Programski jezik je **zapis računanja**.
- Programski jezik je **formalna notacija za natančno opisovanje rešitev problemov**.
- Programski jeziki so najbolj osnovna orodja programerja (T. Hoare).

# Uvod

- Programski jezik mora izpolnjevati **določene zahteve** (D. Watt):
  - Mora biti **univerzalen** (vsak problem mora imeti rešitev, ki jo je mogoče sprogramirati v jeziku, če ta problem sploh lahko rešimo z računalnikom).
  - Mora biti **izvedljiv** na računalniku.
  - Prav tako bi moral biti razumen, **naraven** za reševanje problemov, vsaj problemov znotraj predvidenega področja uporabe.

# Uvod

## ■ Obstajajo različni **načini razvrščanja** in združevanja **programskih jezikov**:

- Na podlagi programskih **paradigm**
  - Proceduralni, funkcijski, logični, objektno usmerjeni, sočasni, ...
- Imperativni vs. deklarativni
  - Imperativno programiranje je osredotočeno okoli **prireditvenih stavkov**, ki omogočajo spreminjanje vsebine celic v pomnilniku.
  - Deklarativno programiranje temelji na **podajanju odnosa med vhodi in izhodi**.
- Namen
  - Poslovna uporaba, umetna inteligenca, sistemsko programiranje, itd.
- Tekstovno (linearno) vs. vizualno

# Uvod

- Obstajajo različni načini razvrščanja ali združevanja programskih jezikov (nadaljevanje):
- Po **generaciji** (1GL, 2GL, 3GL, 4GL, 5GL)
  - 1GL ali jezik prve generacije je strojni jezik.
  - 2GL je zbirnik.
  - 3GL je programski jezik na **visoki ravni** in zahteva veliko znanja o programiranju (C++, JavaScript, uporabljajo prevajalnik ali interpreter).
  - 4GL je zasnovan tako, da je **bližje naravnemu jeziku** kot jezik 3GL (jeziki poizvedb, generatorji poročil, grafični jeziki, aplikacijski generatorji, zelo visoki programski jeziki).
  - 5GL je v bistvu 4GL, ki uporablja sistem, ki **temelji na znanju**.

# Uvod

- Obstajajo različni načini kako razvrstiti programske jezike (nadaljevanje):
  - Nivo abstrakcije
    - Jeziki 1. in 2. generacije: računalniško orientirani (strojni jezik, zbirni jezik)
    - Jeziki 3. generacije: ciljno orientirani (visokonivojski prog. jeziki)
    - Jeziki 4. in 5. generacije: uporabniško orientirani (deklerativni)
  - Splošno namenski (GPL) vs. domensko specifični (DSL)
    - Z GPLji lahko zapišemo programe **za različna področja uporabe**.
    - Z DSLji rešujemo težave na določeni domeni.

# Domensko specifični jeziki

## ■ Definicije:

- Na DSLje je mogoče gledati kot na **programske jezike**, namenjene **določeni domeni** ali problemskemu področju. Zagotavlja ustrezne **vgrajene abstrakcije** in **notacijo**. DSLji so navadno **majhni**, bolj **deklarativni** kot imperativni in manj ekspresivni kot GPLji.

(C. Consel)

- DSL je **programski jezik** ali izvedljiv specifikacijski jezik, ki prek **ustrezne notacije in abstrakcij** ponuja **ekspresivno moč**, osredotočeno in navadno omejeno **na določeno domeno**.

(A. van Deursen, P. Klint, J. Visser)



# Domensko specifični jeziki

## ■ Definicije (nadaljevanje):

- DSL so programski jeziki, ki **žrtvujejo splošnost** za bližino določenega problemskega področja. Z **zmanjšanjem konceptualne razdalje** med problemskim prostorom in jezikom, ki se uporablja za izražanje problema, **programiranje postane preprostejše, lažje in zanesljivejše**. **Količina kode**, ki jo je potrebno zapisati, se zmanjša - poveča se produktivnost in zmanjšajo stroški vzdrževanja.

(Center za Agile Technology –  
Univerza v Teksasu v Austinu)

# Domensko specifični jeziki

## ■ Definicije (nadaljevanje):

- DSL so zasnovani za **ozek obseg aplikacij**. Na splošno se domneva, da se uporabljajo predvsem za zelo majhne programe in so v mnogih primerih namenjeni uporabi za **neprogramerje** (torej "jeziki končnih uporabnikov"). V večini primerov **učinkovitost ni pomembna**, zato je uporabniku jezik prijazen in udoben.

(S. Kamin)

# Domensko specifični jeziki

## ■ Definicije (nadaljevanje):

- DSLji so računalniški jeziki **prilagojeni posebni aplikacijski domeni**. Ponujajo znatne **izboljšave v izraznosti in enostavnosti uporabe v primerjavi z GPLji** na svojem specifičnem področju uporabe.

(M. Mernik, J. Heering, A. Sloane)

## ■ Heering-ov zakon za DSL:

- $\text{izraznost} * \text{velikost domene} = \text{konstantna}$

# Domensko specifični jeziki

- DSLje pogosto poimenujejo tudi drugače:
  - Application-oriented languages
  - Application languages
  - Special-purpose languages
  - Task-specific languages
  - Problem-oriented languages
  - 4GL
  - Specialized languages
  - Little (mini) languages
  - End-user languages

# Domensko specifični jeziki

- Ideja o DSL je verjetno **stara kot programski jeziki**.
- APT (Automatic Programmed Tool), ki je bil razvit v MIT leta 1955, se lahko šteje kot prvi DSL.
  
- Primeri zelo znanih DSLjev:
  - HTML (spletne strani)
  - LATEX (besedilni programi)
  - Make (gradnja programske opreme)
  - SQL (povpraševanje po podatkovnih bazah)
  - VHDL (strojna oprema)
  - BNF (specifikacija sintakse)
  - Excel makro jezik (preglednice)

# Domensko specifični jeziki

- Splošno namenski jeziki (angl. general-purpose languages, GPLji)
  - V kombinaciji z aplikacijsko knjižnico lahko katerikoli GPL deluje kot DSL!
  - Aplikacijski vmesnik knjižnice (API) vsebuje domenski specifični besednjak, ki je sestavljen iz imena razredov, metod in imena funkcij, ki omogočijo, da knjižnica postane dostopna in je možno ustvariti objekte ter opraviti klice metod v kateremkoli programu zapisanem v GPL.
- Zakaj potem razvijati DSLje?
  - Preprosto zato, ker lahko ponudijo dostop do domene na lažji in boljši način.

# Domensko specifični jezik FDL

## GPL

```
boolean norm = true;
//First feature definition ->
// IOdevice : all (opt(Printer), mouse, Display, opt(webcamera), keyboard,
// opt(microphone), opt(Receiver))
FeatureList ioList = new FeatureList(norm);
// opt(Printer)
Feature printerName = RuleManager.buildAtomicFeature("Printer");
Feature printerOpt = RuleManager.buildOptionalFeature(printerName,norm);
ioList = RuleManager.buildFeatureList(ioList,printerOpt,norm);
// mouse
Feature mouseAF = RuleManager.buildAtomicFeature("mouse");
ioList = RuleManager.buildFeatureList(ioList,mouseAF,norm);
// Display
Feature displayName = RuleManager.buildAtomicFeature("Display");
ioList = RuleManager.buildFeatureList(ioList,displayName,norm);
// opt(webcamera)
Feature webcamAF = RuleManager.buildAtomicFeature("webcamera");
Feature webcamOpt = RuleManager.buildOptionalFeature(webcamAF,norm);
ioList = RuleManager.buildFeatureList(ioList,webcamOpt,norm);
// keyboard
Feature keyboardName = RuleManager.buildAtomicFeature("keyboard");
ioList = RuleManager.buildFeatureList(ioList,keyboardName,norm);
// opt(microphone)
Feature micAF = RuleManager.buildAtomicFeature("microphone");
Feature micOpt = RuleManager.buildOptionalFeature(micAF,norm);
ioList = RuleManager.buildFeatureList(ioList,micOpt,norm);
// opt(Receiver)
Feature recName = RuleManager.buildAtomicFeature("Receiver");
Feature recOpt = RuleManager.buildOptionalFeature(recName,norm);
ioList = RuleManager.buildFeatureList(ioList,recOpt,norm);
// IOdevice : ...
Feature ioFeature = null;
Feature ioAllFeature = RuleManager.buildAllFeature(ioList,norm);
ioFeature =
    FeatureDefinition.addFeature(ioFeature,ioAllFeature,
        (AtomicFeature)RuleManager.buildAtomicFeature("IOdevice"));
```

## DSL

```
IOdevice : all (opt(Printer), mouse, Display , opt(webcamera),
               keyboard , opt(microphone), opt(Receiver) )
Printer : one-of (laser , inkjet )
Display : one-of (crt , lcd )
Receiver : more-of(speaker , headphones )
webcamera requires microphone
include lcd
```

```
<...>
Feature printerF = RuleManager.buildOptFeature(printerList,norm);
ioFeature =
```

# Domensko specifični jeziki

## ■ Prednosti:

- DSL omogočajo, da so **rešitve** podane **na ravni abstrakcije domene**.
- Posledično lahko **strokovnjaki iz domene sami** razumejo, preverijo, spremenijo in pogosto celo **razvijajo programe z DSLji**.
- Programi DSL so **berljivi, zgoščeni in razumljivi**, v veliki meri ne potrebujejo veliko dokumentiranja (komentarjev) in jih je mogoče ponovno uporabiti za različne namene.
- DSLji **povečujejo produktivnost**, zanesljivost in prenosljivost.
- DSL **opisujejo domensko znanje** in tako omogočajo ohranjanje in ponovno uporabo tega znanja.
- DSL omogočajo **validacijo in optimizacijo na ravni domene**.



# Domensko specifični jeziki

## ■ Slabosti:

- **Stroški** načrtovanja, implementacije in vzdrževanja DSLjev.
- Stroški **izobraževanja** za uporabnike DSL.
- Omejena **razpoložljivost** DSLjev.
- Težava pri iskanju **območja domene**.
- Težave pri uravnoteženosti med specifičnostjo domene in **splošno namenskimi programskimi konstrukti**.
- Možna izguba **učinkovitosti** v primerjavi z ročno kodirano programsko opremo.

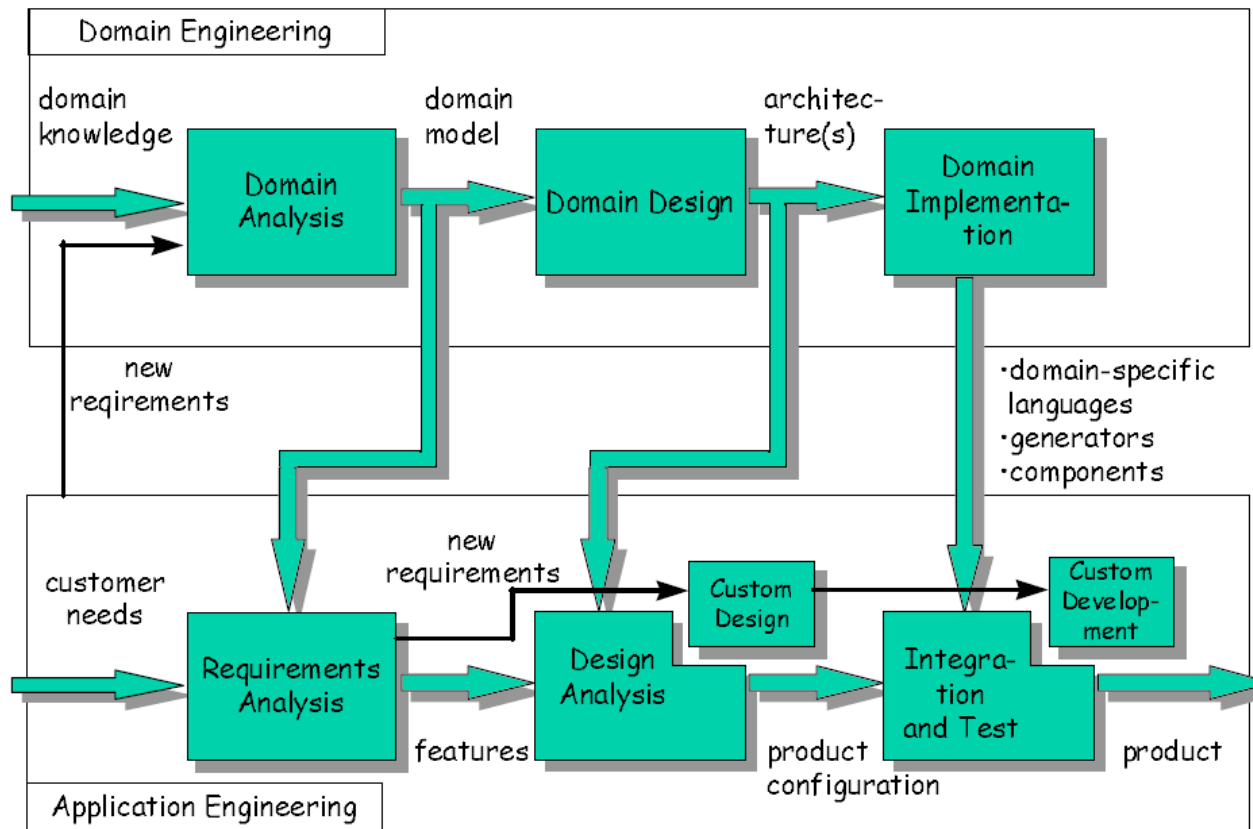
(van Deursen, Klint, Visser)

# DSLji in generativno programiranje

- Generativni razvoj programske opreme je **pristop za sistemsko generiranje družine povezanih rešitev**.
  - Pristop se osredotoča na **avtomatiziranje** ustvarjanja **delov** sistema: sistem se lahko **samodejno generira** iz podanih specifikacij, napisanih **v enem ali več tekstovnih ali grafičnih domensko-specifičnih jezikih**, ki zaokrožujejo značilnosti za posamezno področje (domeno).

(K. Czarnecki)

# DSLji in generativno programiranje



(K. Czarnecki „Generative programming“)

# Povezanost DSMLjev in MDE/MDD

- MDE (model-driven engineering) sestoji iz:
  - **domensko-specifičnih modelirnih jezikov**, ki opisujejo aplikacijsko strukturo in obnašanje,
  - **transformacijskih strojev in generatorjev**, ki analizirajo modele in generirajo različne programske artefakte (npr. programsko kodo, dokumente, simulacijske skripte, XML namestitvene skripte).

(D. Schmidt)

# DSLji in produktne linije

- Ideja produktnih linij (angl. software product lines):
  - **Gradimo en sistem**, ki bo sposoben **generirati maso** „majhnih“ programskih variant
- Programske produktne linije so tehnologija, ki omogoča **konfiguriranje za množično proizvodnjo programske opreme**. Konfiguratorji produktnih linij vzamejo dve vrsti vhodov: **osnovne programske dele** in **produktne modele** z namenom avtomatičnega generiranja programskega produkta.

(C. Krueger)

# DSLji in programske tovarne

## ■ Programske tovarne

- Programske tovarne (angl. software factories) **je razvojno okolje namenjeno razvoju specifičnega tipa aplikacij.**

## ■ Programske tovarne uporabljajo:

- **domensko specifične jezike** (DSLje), ki eksplicitno predstavijo domenske koncepte in
- **domenska orodja**, ki enkapsulirajo znanja o implementaciji.

(Greenfield et al.)

# Predmet „Domensko specifični modelirni jeziki“

- DSL-ji so bili uporabljeni **na različnih področjih** in njihova uporaba je jasno pokazale prednosti DSL-jev pred sorodnimi rešitvami
  - **produktivnost, zanesljivost, vzdržljivost in fleksibilnost.**
- Vendar pa je potrebno **upoštevati stroške za načrtovanje, razvoj in vzdrževanje DSLjev**. Brez ustrezne metodologije in orodij so ti stroški lahko višji od prihrankov, pridobljenih s kasnejšo uporabo DSLjev.
- V tem predmetu bomo spoznali ustrezne **metodologije in orodja**, ki so potrebni za podporo razvoja DSL-jev.

# Vprašanja

