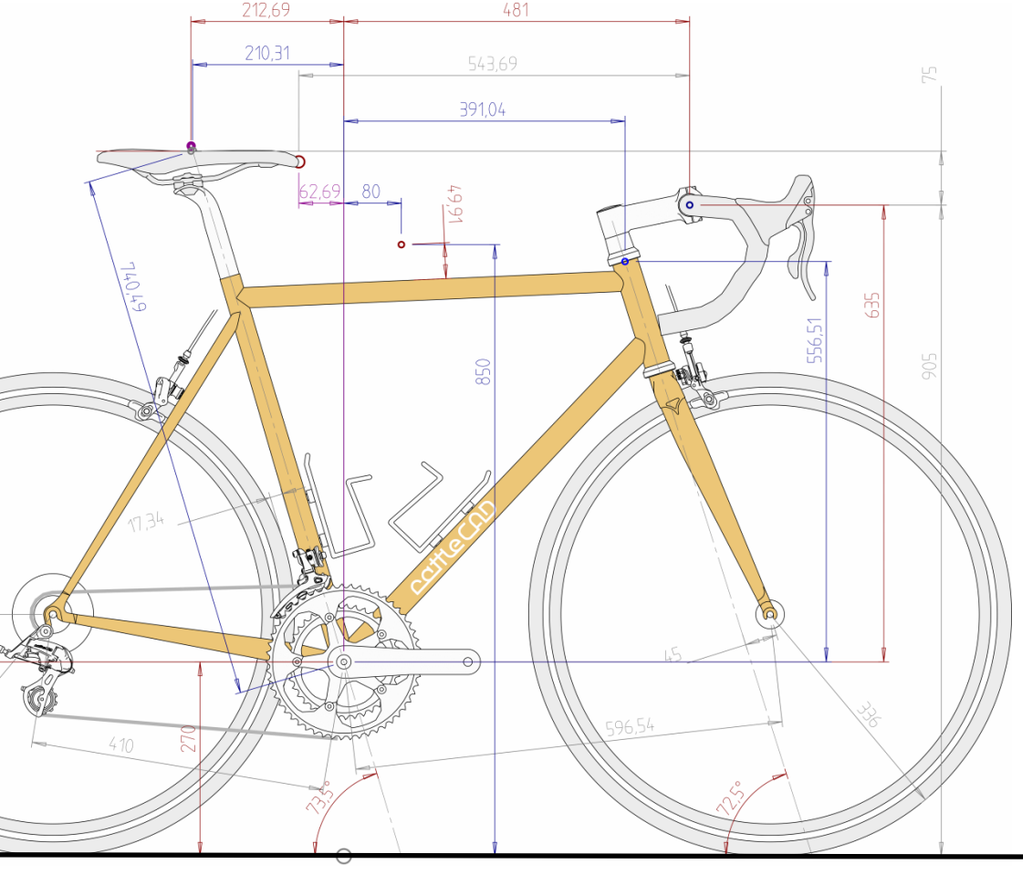


15th European Tcl/Tk User Meeting

July, 8th and 9th 2017, Berlin, Germany

RattleCAD

http://rattlecad.sourceforge.net/

… design your custom Bike

Manfred Rosenberger

Welcome

rattleCAD

- what is it for

refactor rattleCAD

- initial situation
  - history & architectural decisions
- solution approach:
  - Mapping Table
  - Strategy Pattern

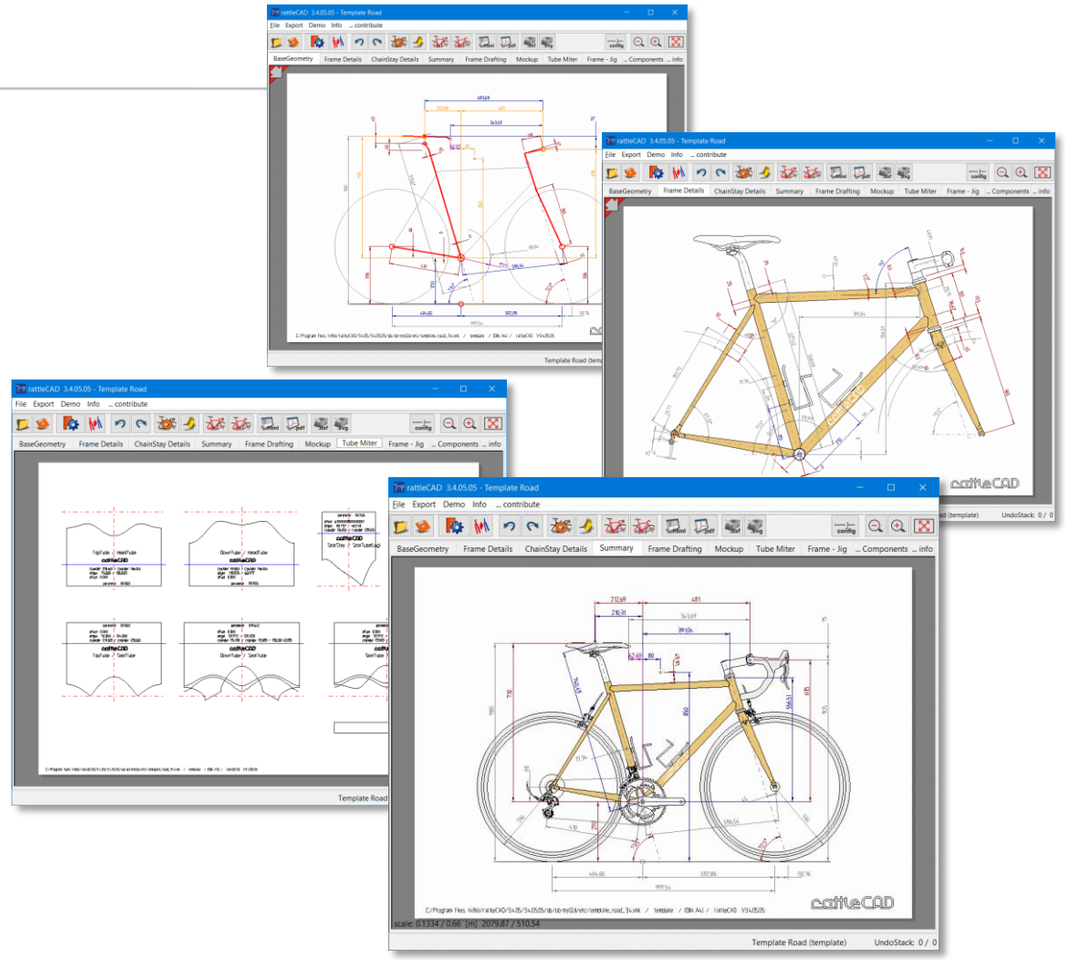
canvasCAD

- the CAD perspective

Tcl/Tk Wishlist

- features, documentation

Let‘s talk …



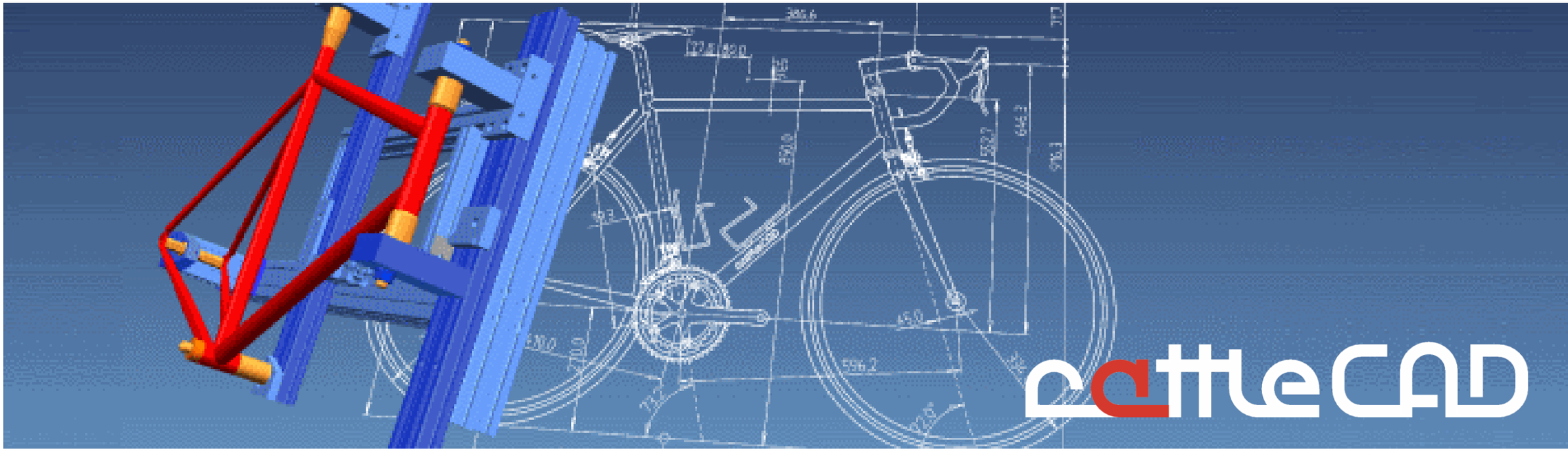
Manfred ROSENBERGER MSc

- Requirements Engineering
- CAD & PDM Support
- licence to teach in secondary schools

- Companies work(ed) for
  - Virtual Vehicle, Senior Researcher (2008 - …)
  - MAGNA STEYR Fahrzeugtechnik (1999 – 2008)
  - TB Schimpel (1998-1999)
  - HOCO - Windows

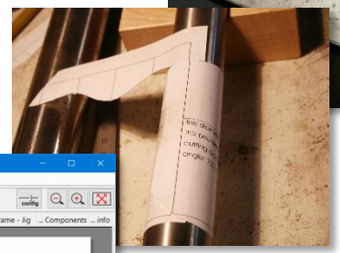
manfred.rosenberger@gmx.net

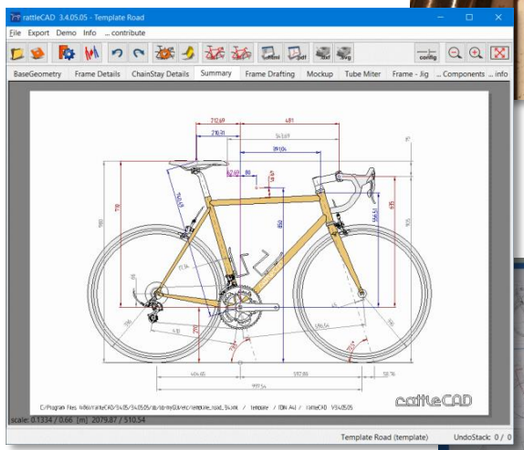
… may I introduce

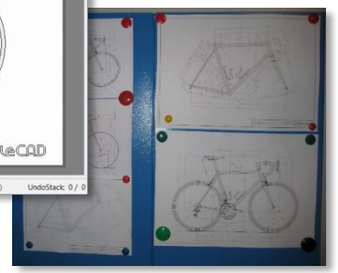


Main Features

- **support bespoked bicycle frame builders**
- **guide through the design process**
- **create workshop drawings**
  - main miters of round tubes
  - settings for different frame jigs
- **build a mockup**
- **different export formats**
  - HTML
  - SVG
  - DXF
  - PDF

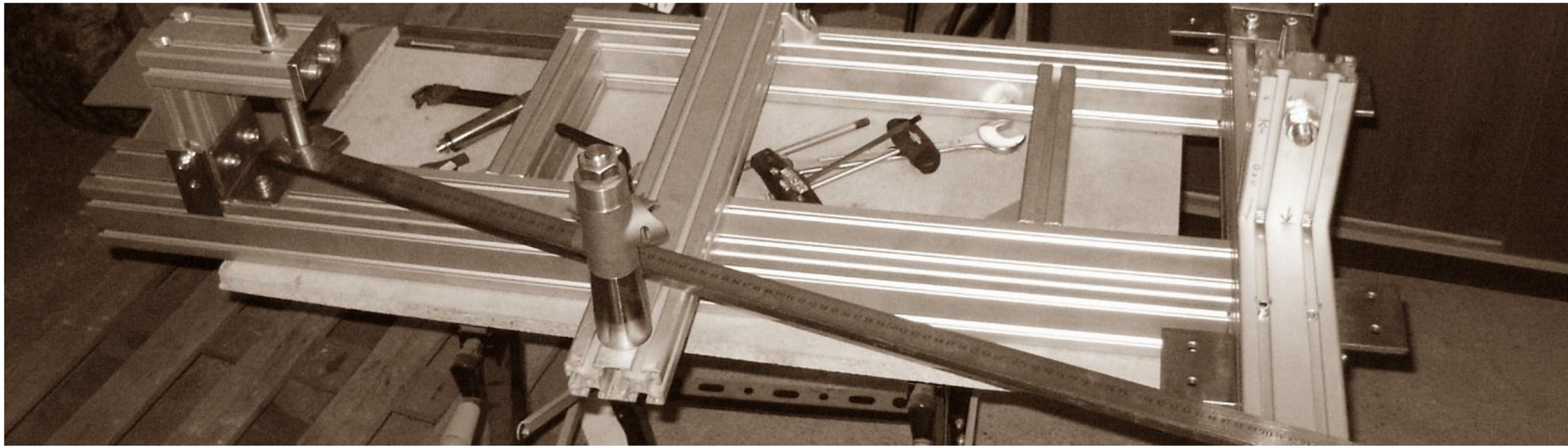




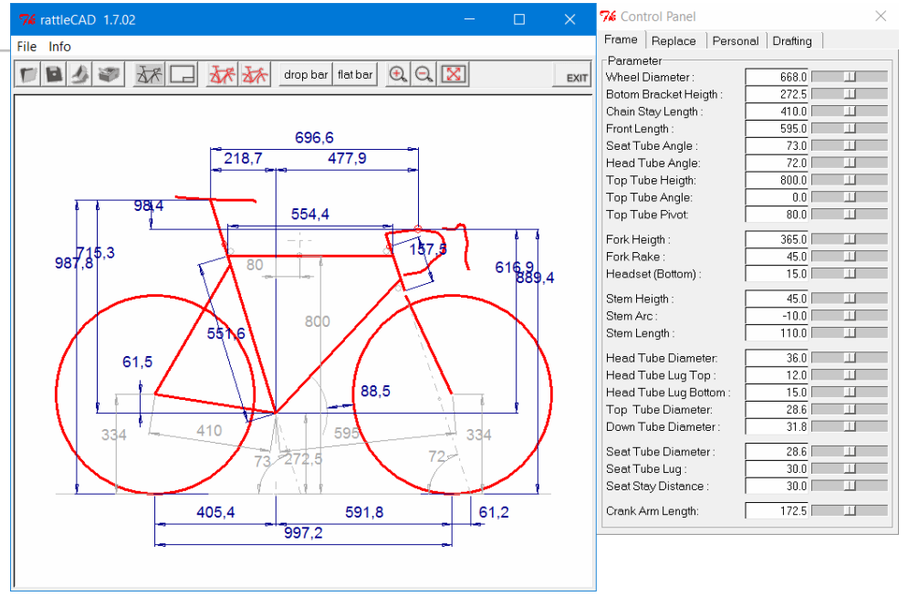


local

refactor rattleCAD …



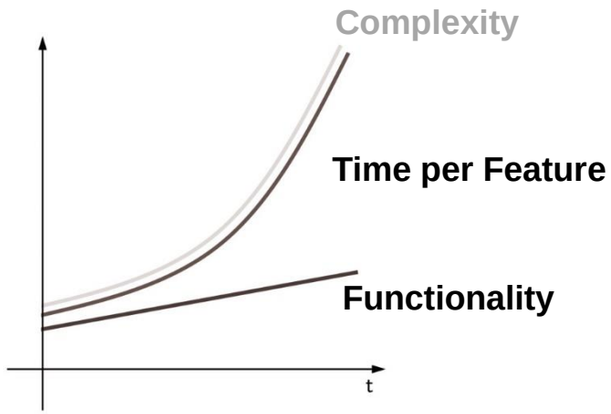
History: Why



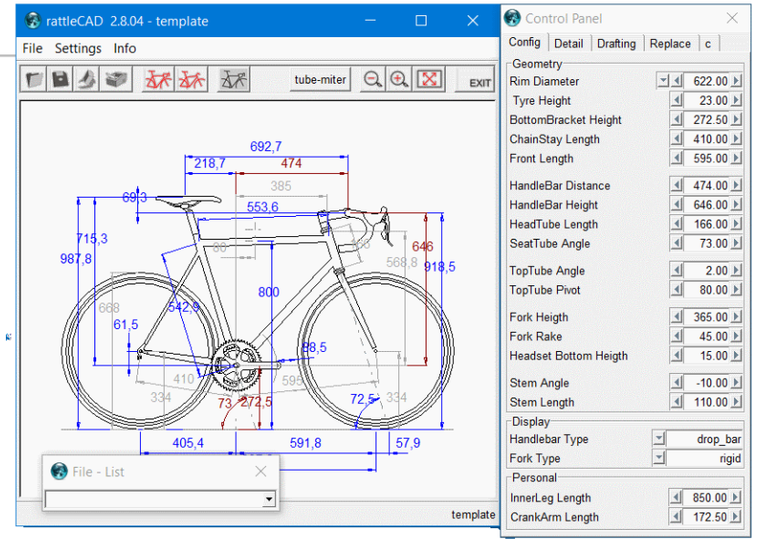
- **first motivation**
  - playing with the canvas
  - deal with bicycle geometry (context)
  - „how to“ and features in focus
  - a prototype in a sandbox
- **Conditions**
  - no architectural approach
  - no professional development environment
  - no plan for a professional software development
- **TIP**
  - dont forget to exit the experiment

History: The experiment becomes an application

- **Situation**
  - forgotten to exit the experiment
  - a lot of effort allready in the application
  - new features to implement
- **Condition**
  - architecture is not ready for new features



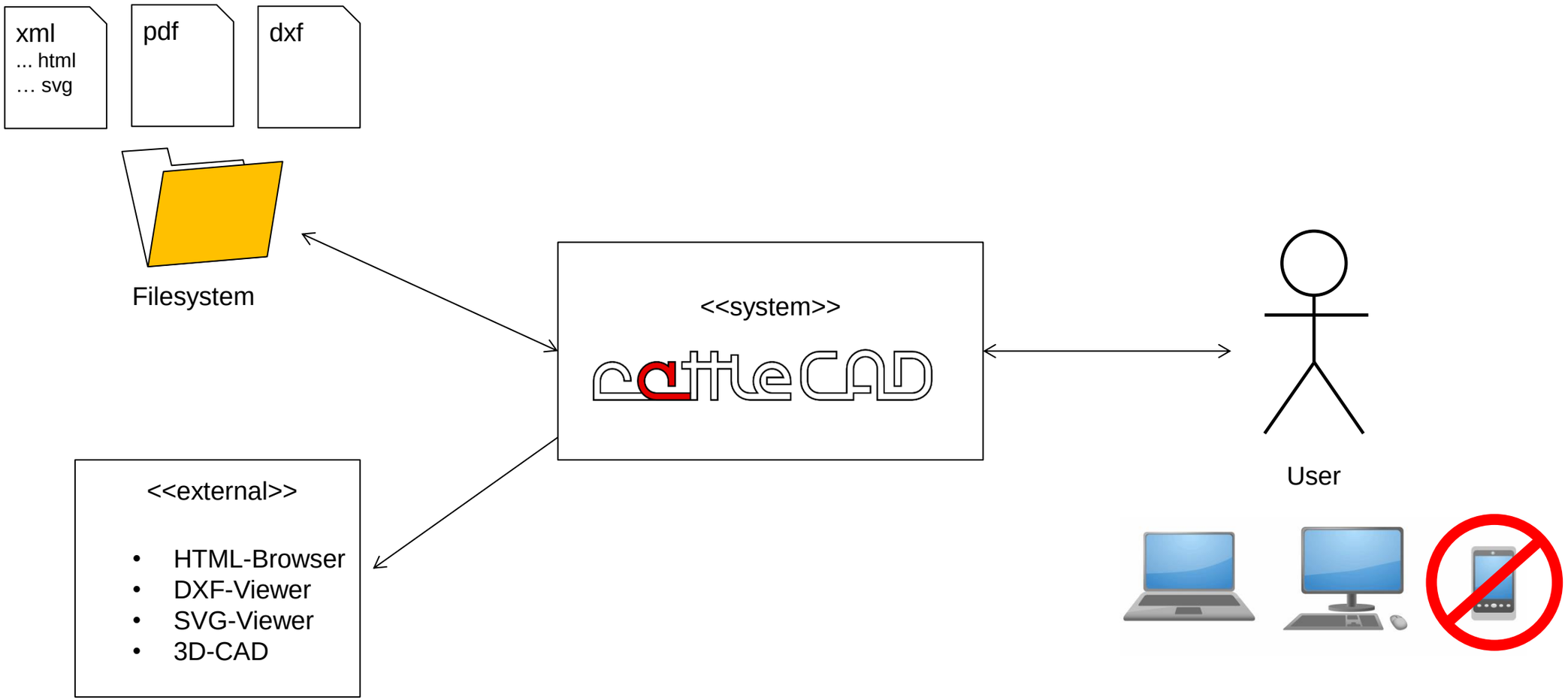
Graphics: http://www.elektronikpraxis.vogel.de



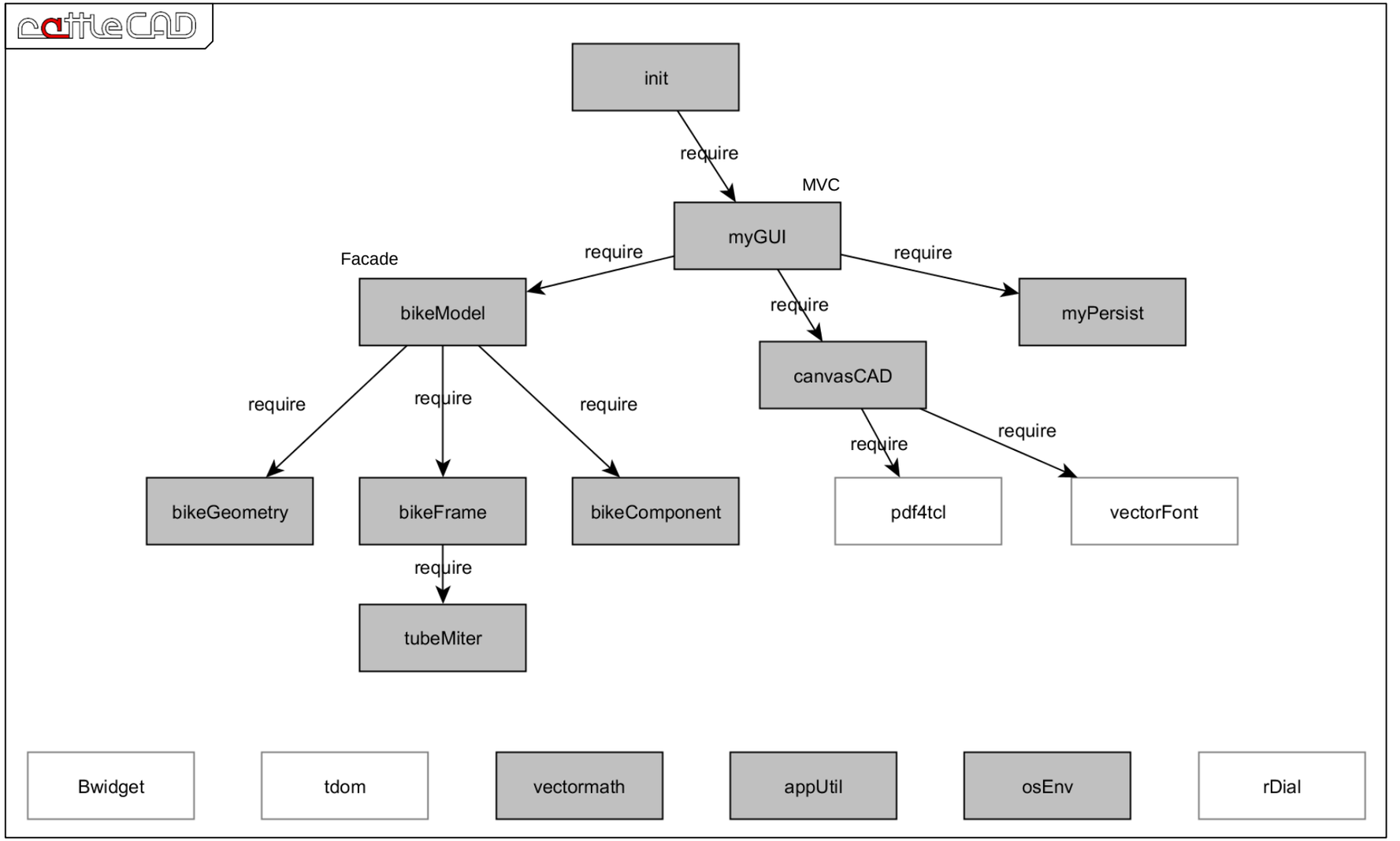
Architectural decisions:

- **packages only from the default Tcl/Tk environment**
- **exceptions as little as possible**
  - **tdom**
  - **pdf4tcl**
  - **rdial**
  - **vectorfont**

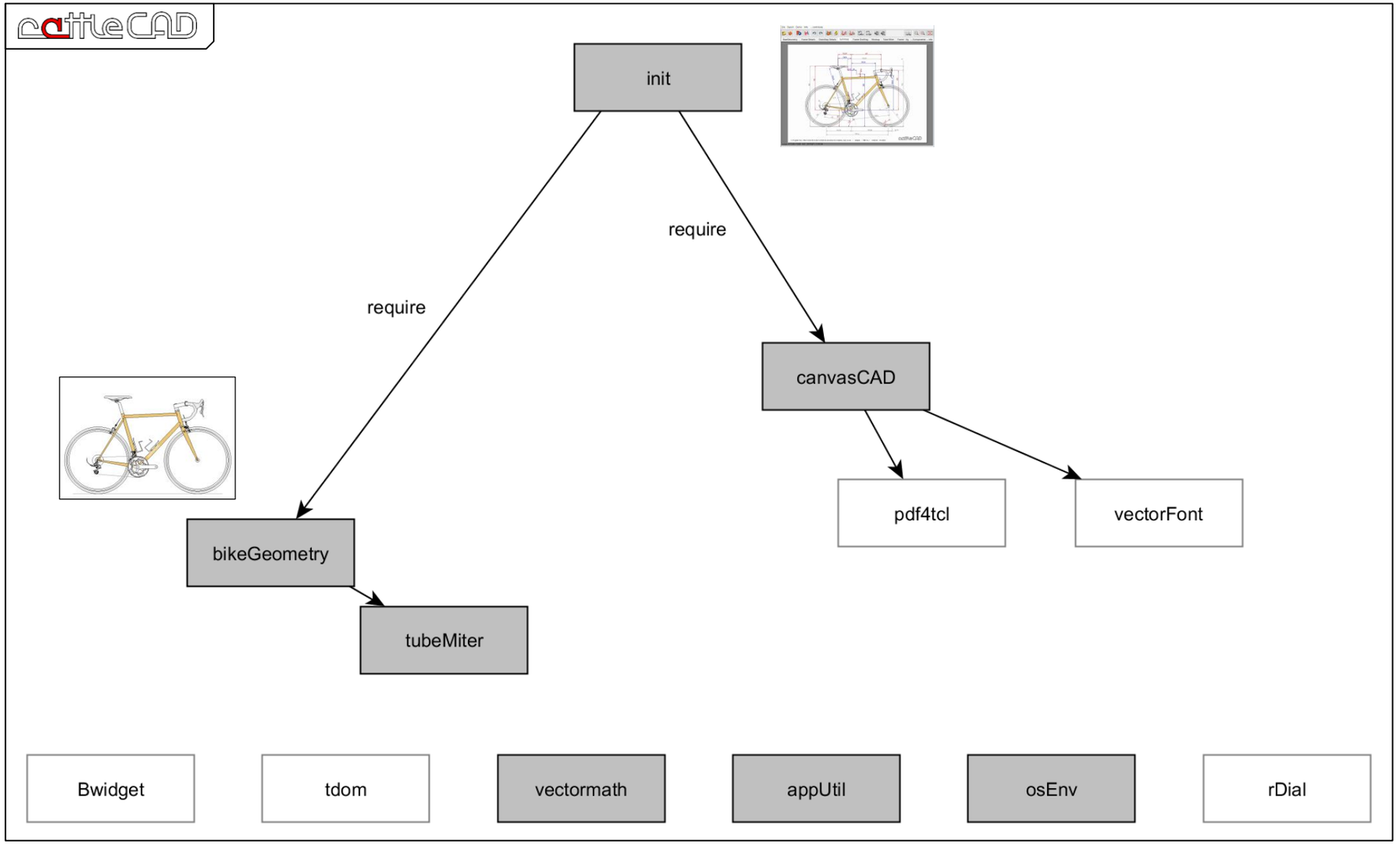
System Analysis: Context



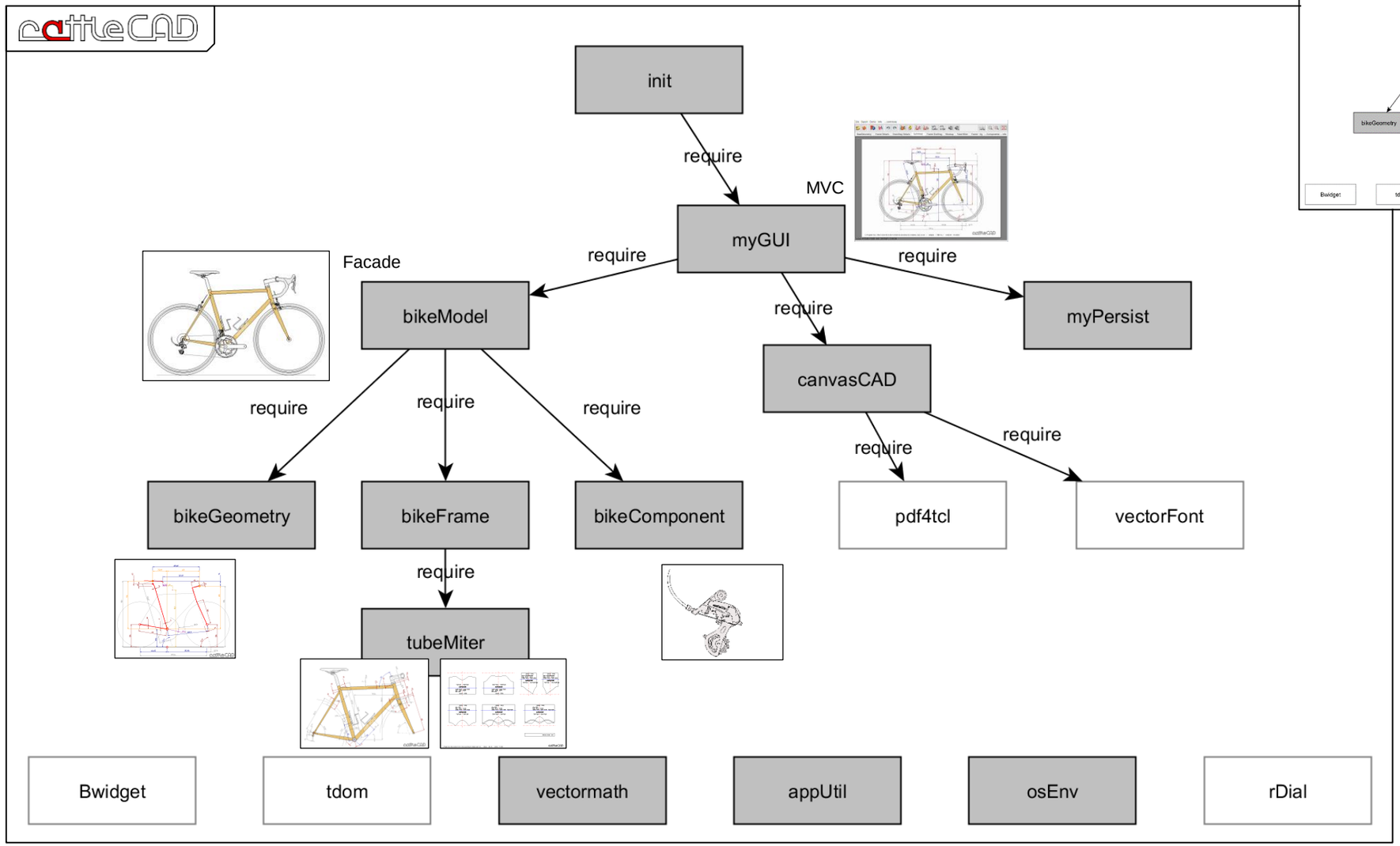
System Analysis: Packages (target Situation)

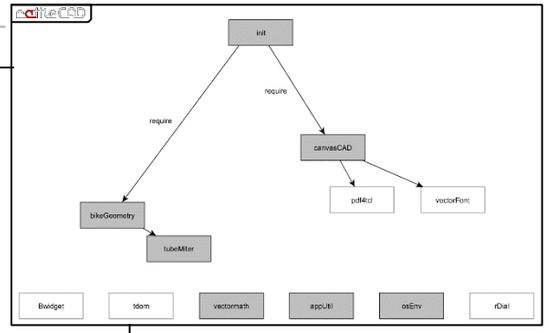


System Analysis: Packages (initial Situation)

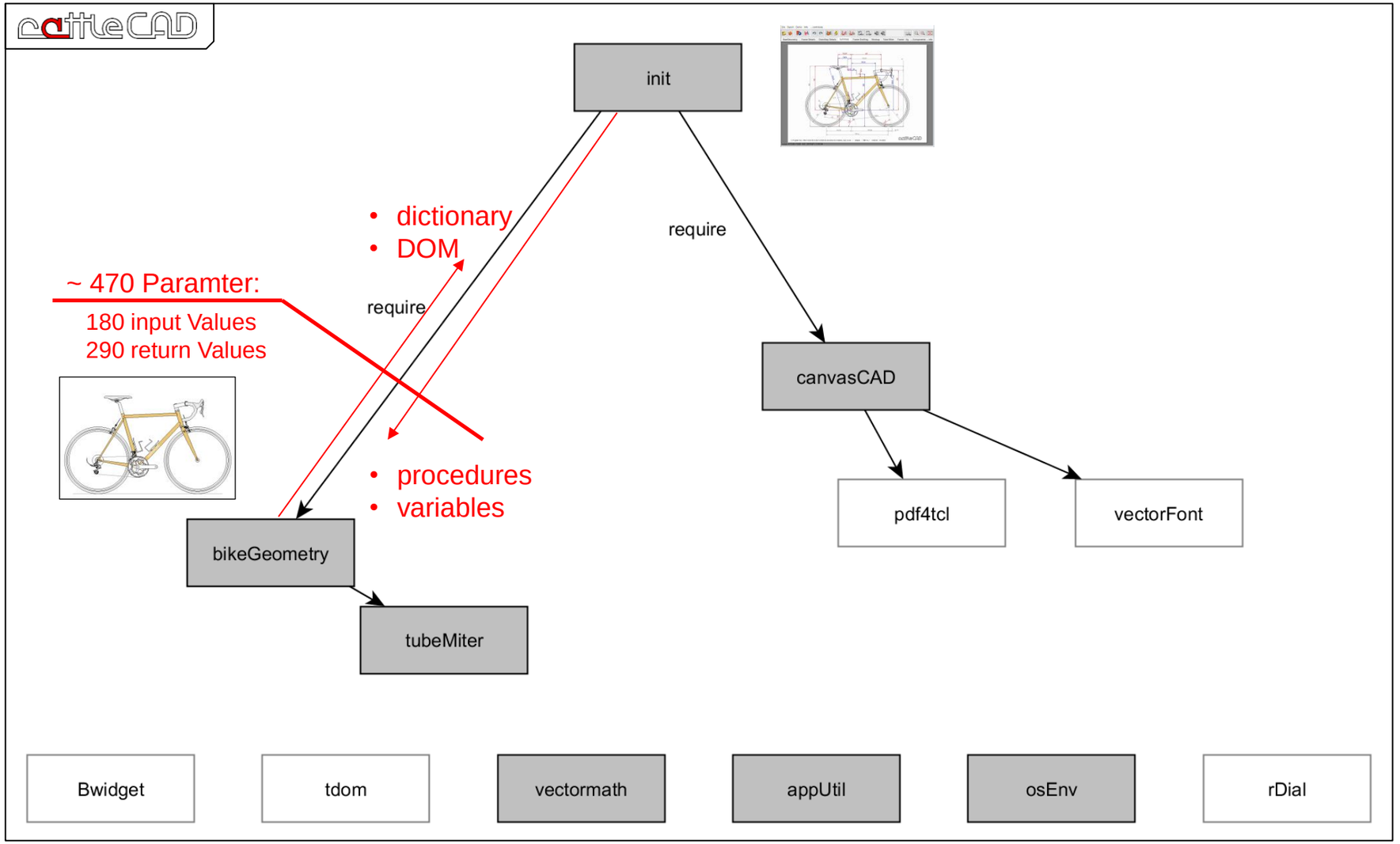


System Analysis: Packages (target Situation)

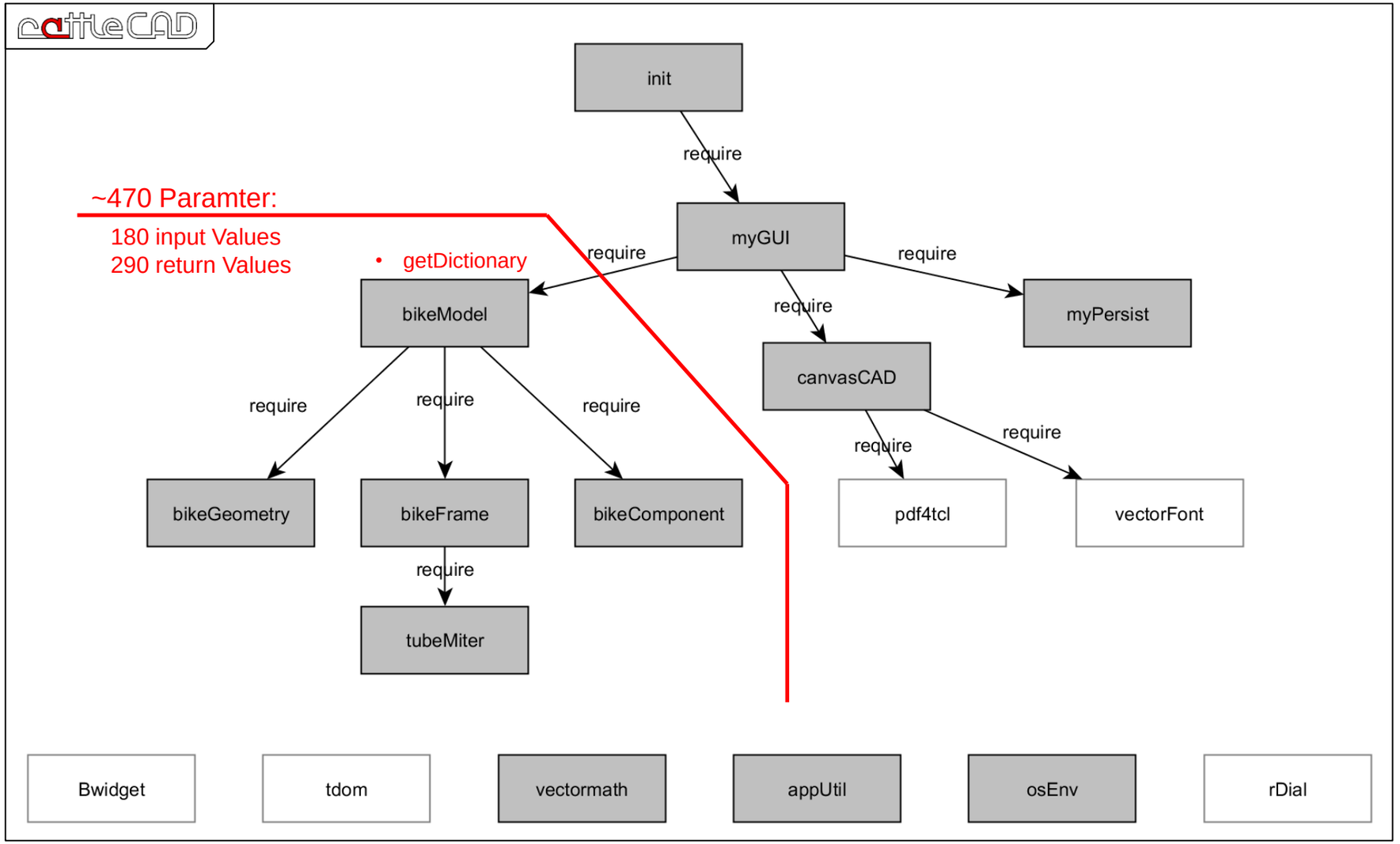




System Analysis: Packages (initial Situation)



Solution Approach: Encapsulation



Solution Approach: encapsulate bikeModel

~470 Paramter:

- **180 input Values**
- **290 return Values**

bikeModel

- setScalar (pathName)
- setConfig (pathName)
- getDictionary

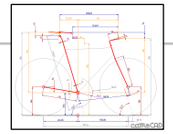
1

n

childObject

- array Scalar ()
- array Config ()
- array Position ()
- array Polyline ()
- array ComponentNode ()

Approach: Mapping Table



bikeGeometry

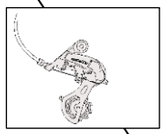
- setScalar (pathName)
- setConfig (pathName)
- getDictionary

- CrankSet
- HeadTube
- Stem
- …

bikeFrame

- setScalar (pathName)
- setConfig (pathName)
- getDictionary

- DownTube
- HeadTube
- SeatTube
- TopTube
- …



bikeComponent

- setScalar (pathName)
- setConfig (pathName)
- getDictionary

- CrankSet
- Derailleur
- FrontBrake
- RearBrake
- Stem
- …

- DownTube
- HeadTube
- RearBrake
- FrontBrake
- Stem
- …

Solution Approach: Mapping Table

```
<root>
  <ObjectName>
    <VariableType>
      <VariableName
          init=„0/1“
          geometryKey=„keyToValue“
          frameKey=„keyToValue“
          componentKey=„keyToValue“
      />
    />
  />
</root>
```

```
<root>

    <.../>

    <HeadTube>
        <Config>
            <Type init="1" geometryKey="" frameKey="HeadTube/Config/Type" component
        </Config>
        <Direction>
        <Polyline>
        <Polygon>
            <Shape_XZ init="0" geometryKey="" frameKey="HeadTube/Shape/xz" componer
        </Polygon>
        <Position>
        <Profile>
        <Scalar>
            <Diameter init="1" geometryKey="" frameKey="HeadTube/Scalar/Diameter" c
            <DiameterTaperedBase init="1" projectXML="HeadTube/DiameterTaperedBase"
            <DiameterTaperedTop init="1" projectXML="HeadTube/DiameterTaperedTop" g
            <HeightTaperedBase init="1" projectXML="HeadTube/HeightTaperedBase" geo
            <Length init="1" geometryKey="Scalar/HeadTube/Length" frameKey="HeadTub
            <LengthTapered init="1" geometryKey="" frameKey="HeadTube/Scalar/Length
        </Scalar>
        <Vector>
    </HeadTube>

    <.../>

</root>
```

Solution Approach: Mapping Table

```
<root>
  <ObjectName>
    <VariableType>
      <VariableName
          init="0/1" geometryKey="{}/keyToValue" frameKey="{}/keyToValue" componentKey="{}/keyToValue"/>
```

1 … input parameter
0 … result parameter

{} … package does not provide this parameter

```
bikeModel::setScalar HeadTube/Length "165"
```

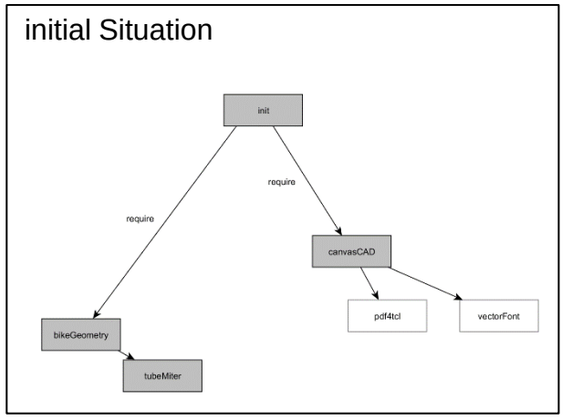
```
<HeadTube>
  <Scalar>
    <Length init="1" geometryKey="Scalar/HeadTube/Length" frameKey="HeadTube/Scalar/Length" componentKey=""/>
```

bikeModel::update → bikeGeometry::update → bikeFrame:: update → bikeComponent:. update

```
    <Shape_XZ init="0" geometryKey=""                    frameKey="HeadTube/Shape/xz"      componentKey=""/>
  <Polygon>
<HeadTube>
```

```
bikeModel::getDictionary
```

Solution Approach: Step by Step



create new package: myGUI
… transfer features
from package: init

1

cut all links to package: bikeGeometry
and link to package: bikeModel

4

create new package: bikeModel
… with well designed
data structure

2

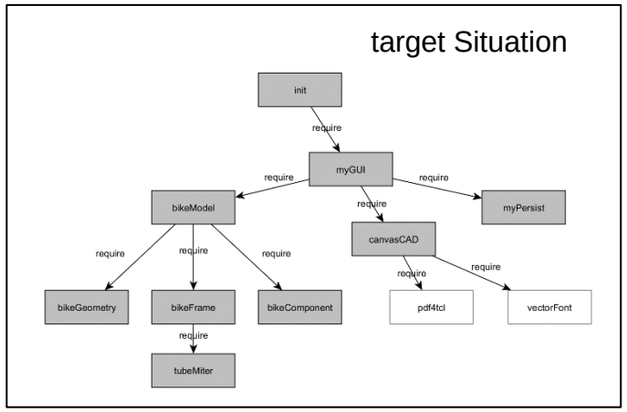
3

map input and result data
to package: bikeGeometry
using the mapping table

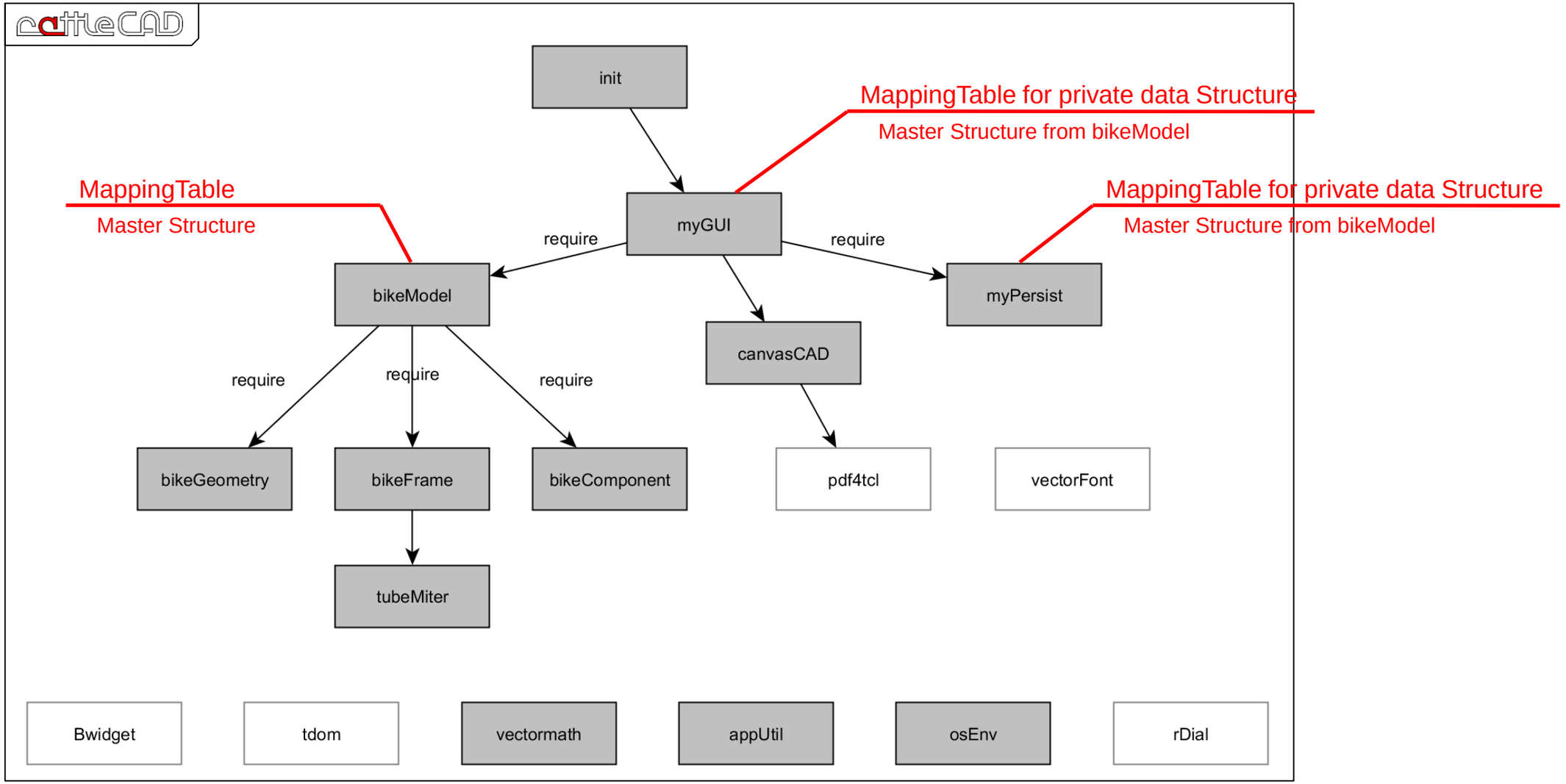
5

extract packages:
bikeFrame and bikeComponent
cleanup package:
bikeGeometry

intermediate steps



Solution Approach: Extended Use of Mapping Tables

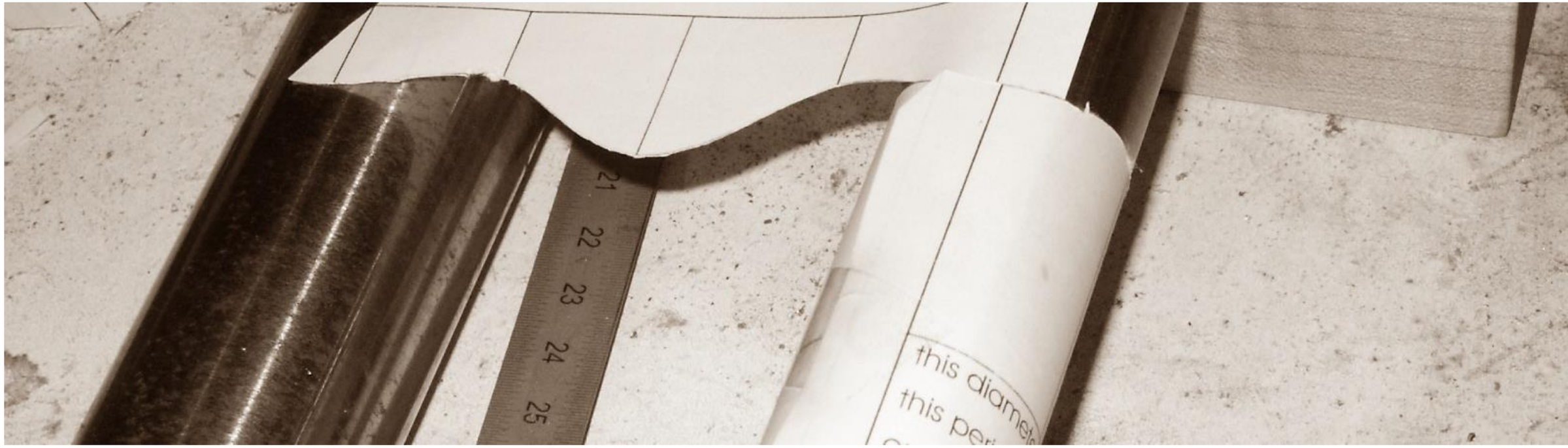


Solution Approach: Mapping Table - Summary

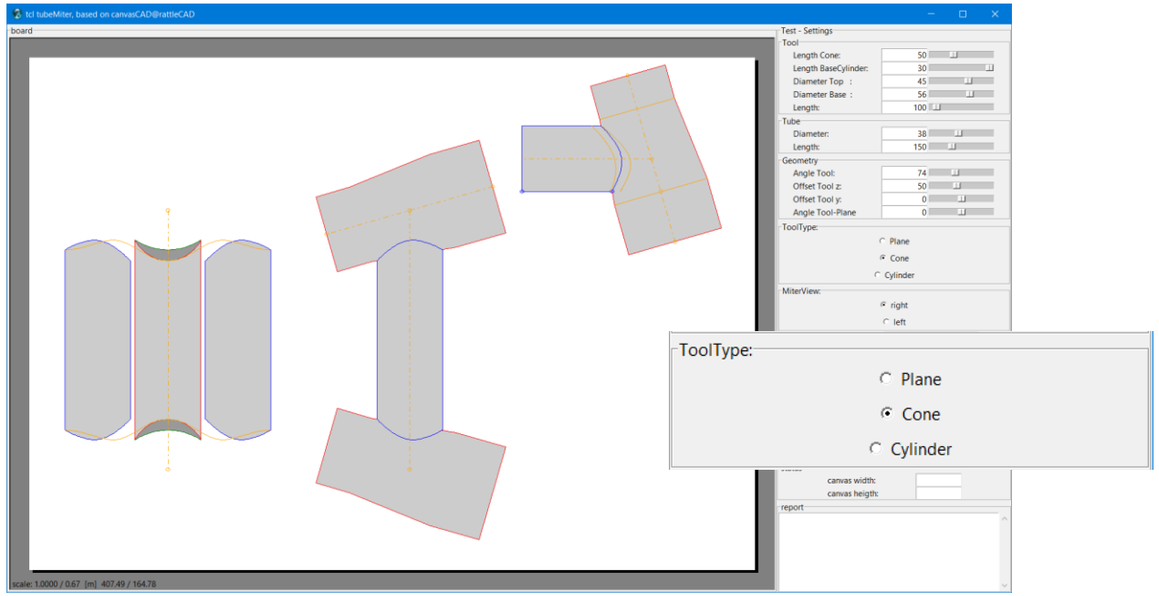
```
<root>
  <ObjectName>
    <VariableType>
      <VariableName
          init=„0/1“
          geometryKey=„keyToValue“
          frameKey=„keyToValue“
          componentKey=„keyToValue“
      />
    />
  />
</root>
```

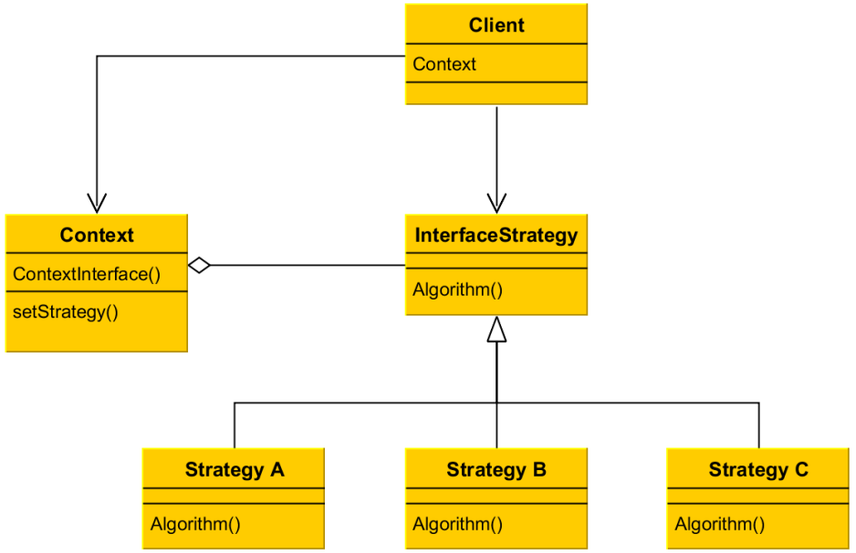
- **package: bikeModel**
  - returns dictionary based on mapping table
  - mapping table defines the master structure for all using packages
- **packages, using the result dictionary provided by bikeModel**
  - have its own mapping table for internal use
  - mapping tables follow the master structure from bikeModel
- **Benefit**
  - dict transports SVG-DOM Objects (tdom)
  - refactoring
    - common procedures/methods keep parameters and values current
  - debugging
    - compareable mapping tables for debugging purpose

refactor tubeMiter (package) …



refactor tubeMiter with TclOO

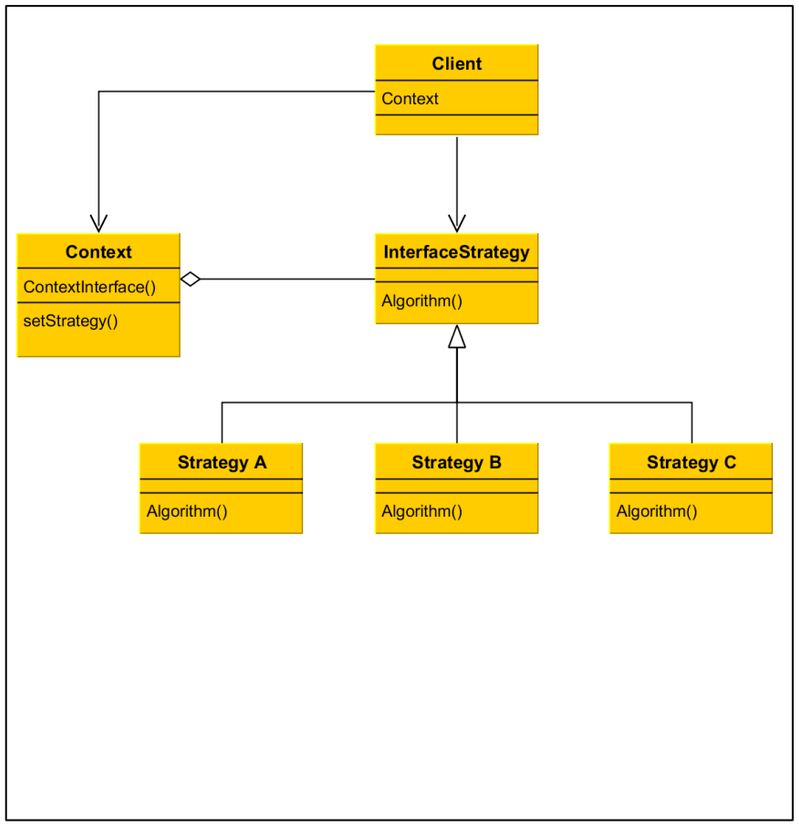


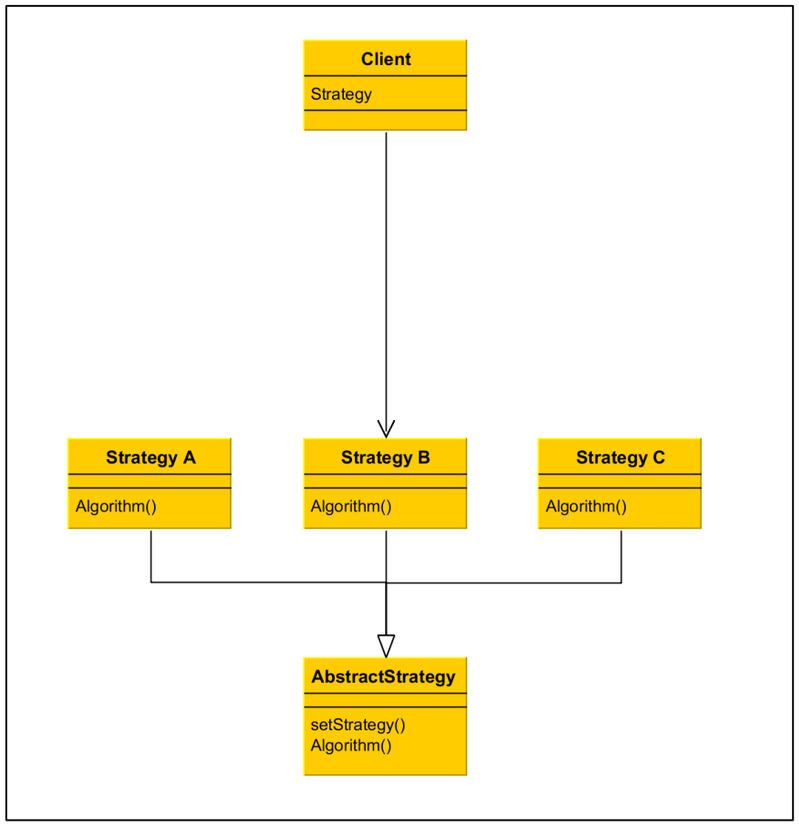


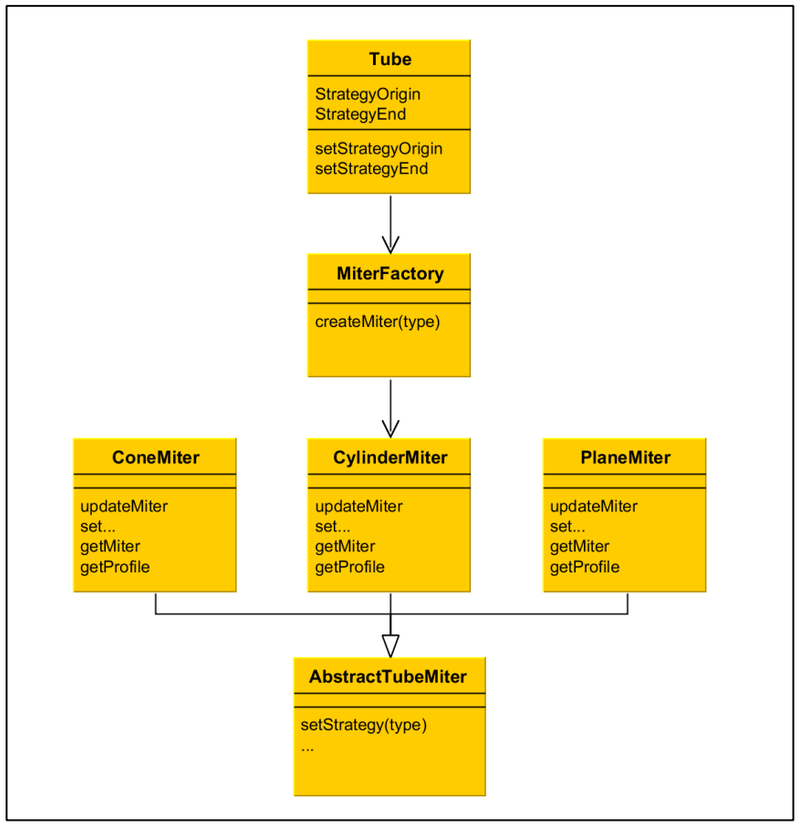
https://de.wikipedia.org/wiki/Strategie_(Entwurfsmuster)

http://www.magicsplat.com/articles/oo.html

Strategy Pattern with TclOO





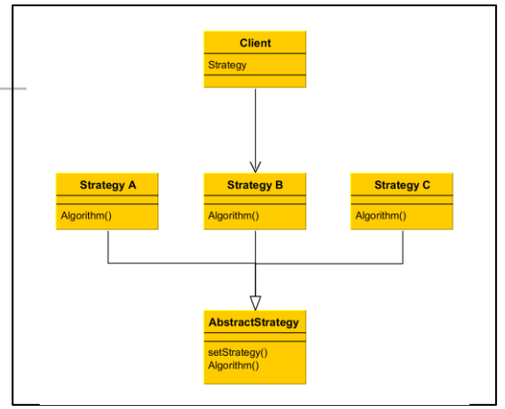


https://de.wikipedia.org/wiki/Strategie_(Entwurfsmuster)

Tcl with TclOO

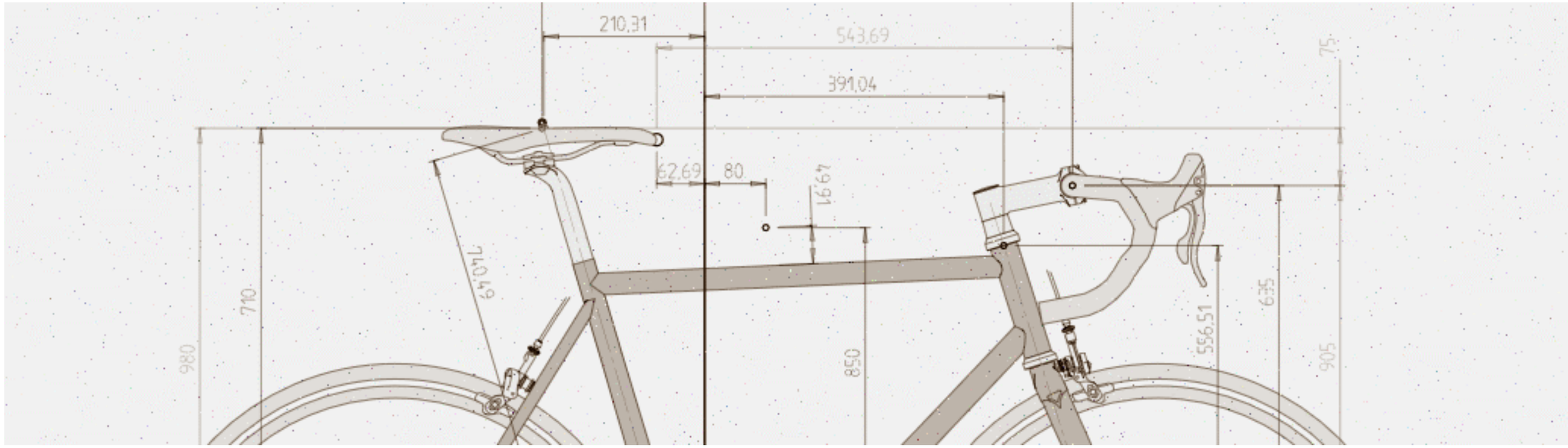
implementation in cattleCAD

Solution Approach: TclOO – Summary (Strategy Pattern)



- **divide and conquer (simpler subproblems)**
- **hide complexity**
- **understandable**
- **testable**
- **…**

… canvasCAD – the CAD perspective



canvasCAD: Requirements & Feature List

Tk

canvas

- **Origin: top-left**

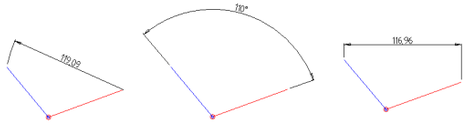
- **File formats**
  - PS (export)

cattleCAD

canvasCAD

- **Origin: bottom-left**

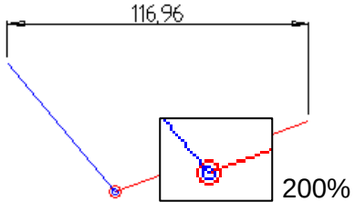
- **Default scale (eg. 1:5)**
- **pan & zoom**
- **Dimensioning**



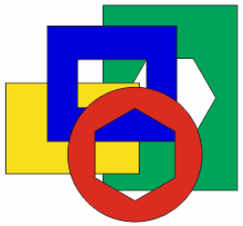
- **File formats**
  - SVG (import & export)
  - DXF (export)
  - PDF (export)

missing in Tk::canvas

- **antialiasing**



- **fully SVG-compatibility**
  - e.g.: path-Element



Wish-list to the TCL/Tk-Community

Wishlist I – (Tk 9.0 WishList - https://wiki.tcl.tk/4055)

- **canvas**
  - 74. Anti-aliased canvas
  - 20. Rotation of photos
  - 80. Oval rotation
  - 81. Zooming
    - … including images on the canvas
  - 115. canvas
- **other**
  - 22. Transparent color
  - 58. Background Images And Transparency
  - 97. Combine ttk::treeview with a listvariable
  - 69. Improved bitmap editing

Wishlist II – (Tk 9.0 WishList - https://wiki.tcl.tk/4055)

- **installer Framework, to deploy applications**
  - standard user are no computer experts (especially Mac-user)
  - starpack is not enough
    - MacOS requires an installation package
- **Documentation**
  - wiki.tcl.tk …
    - link to the manuals (https://www.tcl.tk/man) on a more prominent position
  - description:
    - textual description is necessary, but often difficult to understand
    - an abstract to explain what a library is for
      - Context, Challenge, Approach and an Example of usage
  - examples:
    - one running script, showing the main purpose of a command or library (like Bwidget)
    - one running script per provided feature
      - what about the test-procedures of the provided libraries … dont they exist?

Summary

Summary

- rattleCAD
- Mapping table to handle parameters in different packages
- use of the Strategy-Pattern with TclOO
- use of canvas in a CAD-Environment
- Wish-list of features to the TCL/Tk-Community

- let‘s talk …

Manfred Rosenberger
manfred.rosenberger@gmx.net