



# Software Tools and Data Formats for Data Exchange in Airplane Predesign

Markus Litz, Holger Cornelsen, Hans-Peter Kersken  
Simulation and Software Technology  
German Aerospace Center (DLR)

PDE 2008

Noordwijk, The Netherlands

March 27<sup>th</sup>, 2008



Deutsches Zentrum  
für Luft- und Raumfahrt e.V.  
in der Helmholtz-Gemeinschaft



# Overview

- Motivation and Background
- Common Data Format
- Software Tools
- Integration Framework

# Motivation and Background



- The predesign of new airplane configurations involves many different technical disciplines
- Goal: Find an optimal design
- Strong dependencies exist between the disciplines
  - A combination of discipline-local optima does not lead to a global optimum
  - Necessary: global optimization process
    - ➔ Look at the overall system
- Therefore: Cooperation between the individual technical disciplines is essential



# Situation at DLR

- Many aerospace institutes, each one specialized on its own technical discipline
- Simulation software is institute-specific, proprietary I/O formats
- Interdisciplinary cooperation in some cross-institutional projects

## But:

- Ad-hoc definition of interfaces and data formats in each project
- No common data format for all application codes
- No automated process chains
  - Applications used manually and separately from each other



# Linking of Discipline-Specific Design Tools

## ➤ Goal

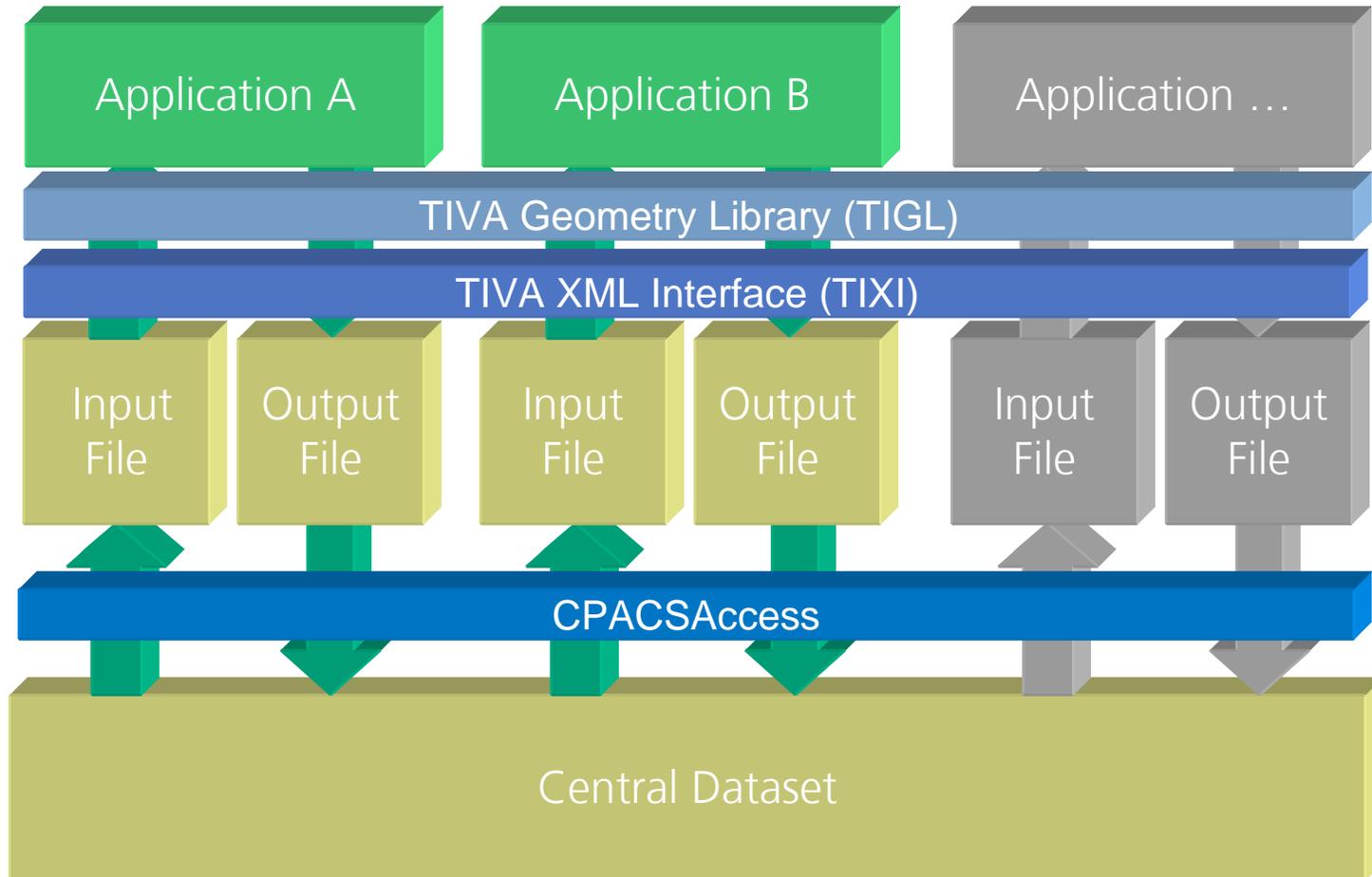
A DLR-wide system to enable the multi-disciplinary design and analysis of airplane configurations in the predesign phase.

## ➤ Under development at DLR in the following aeronautics projects:

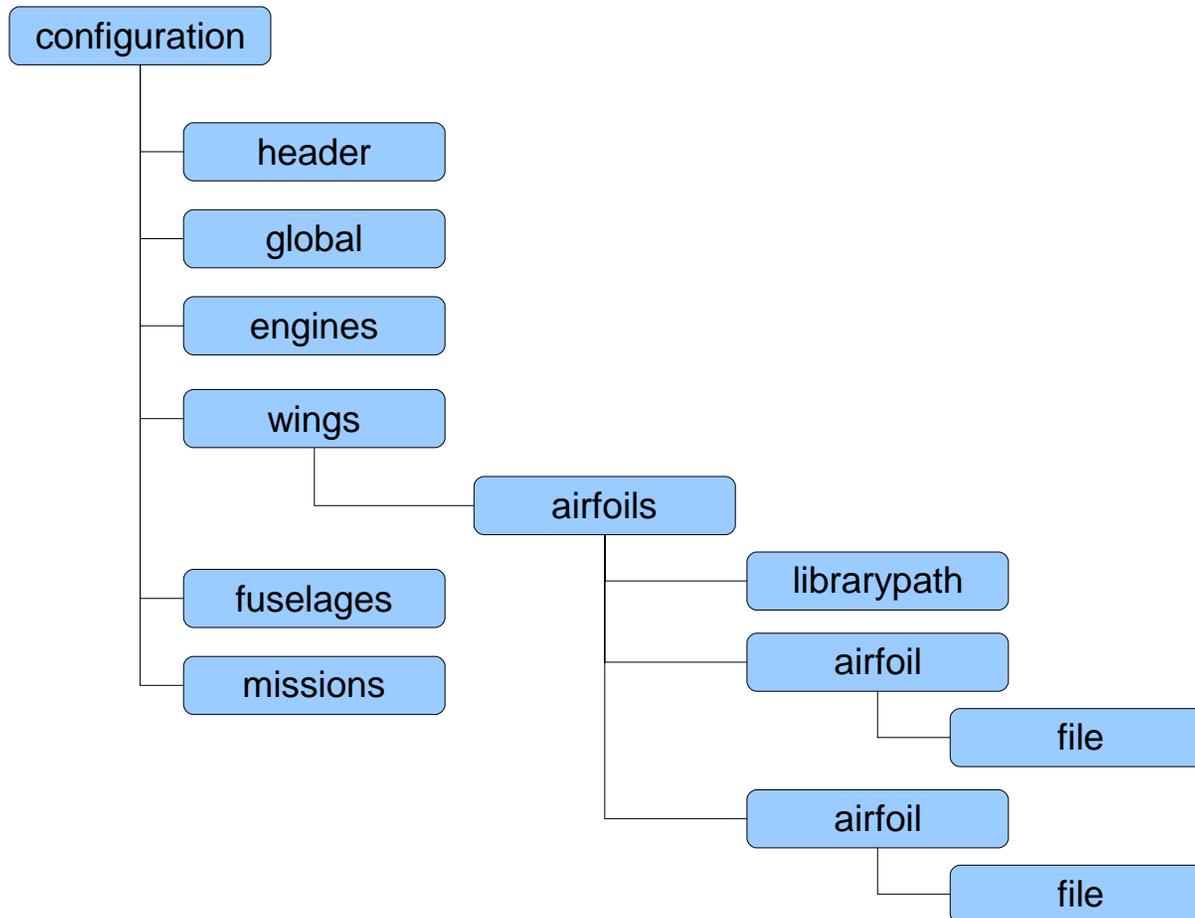
- TIVA I/II – Technology integration for the virtual aircraft
- UCAV 2010 – Unmanned combat air vehicle
- EVITA – Evaluation of innovative turbine engines

## ➤ Similar requirements in DLR space projects

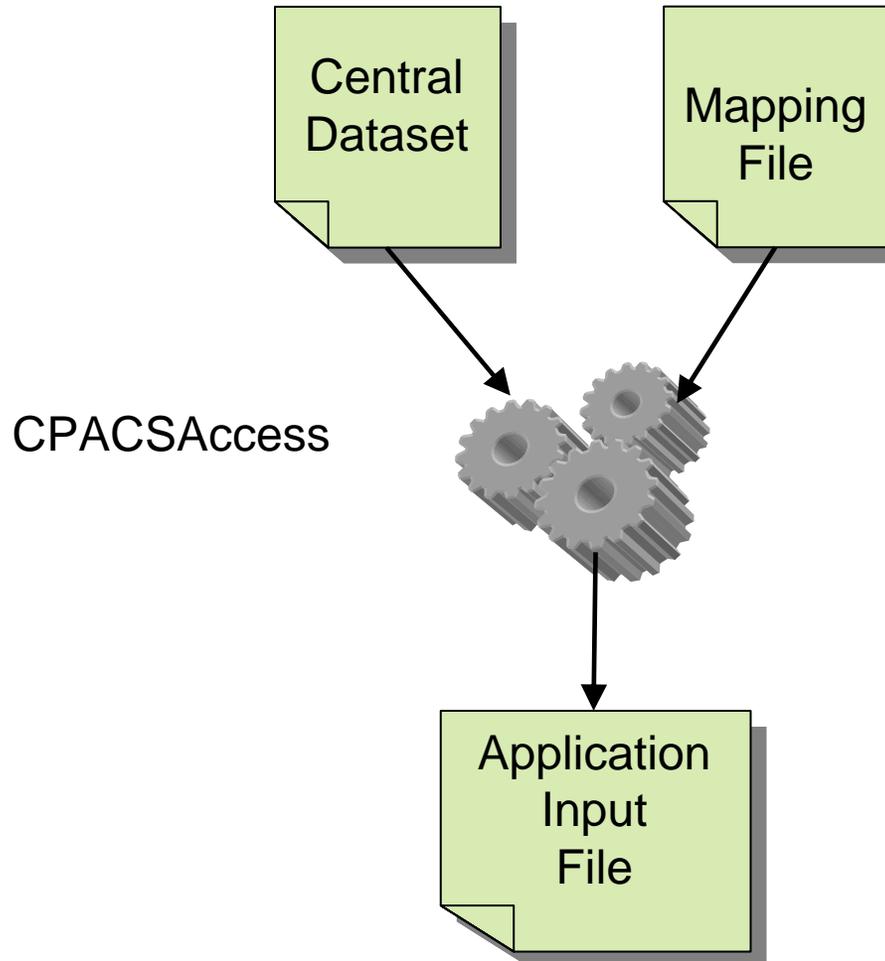
# System Overview



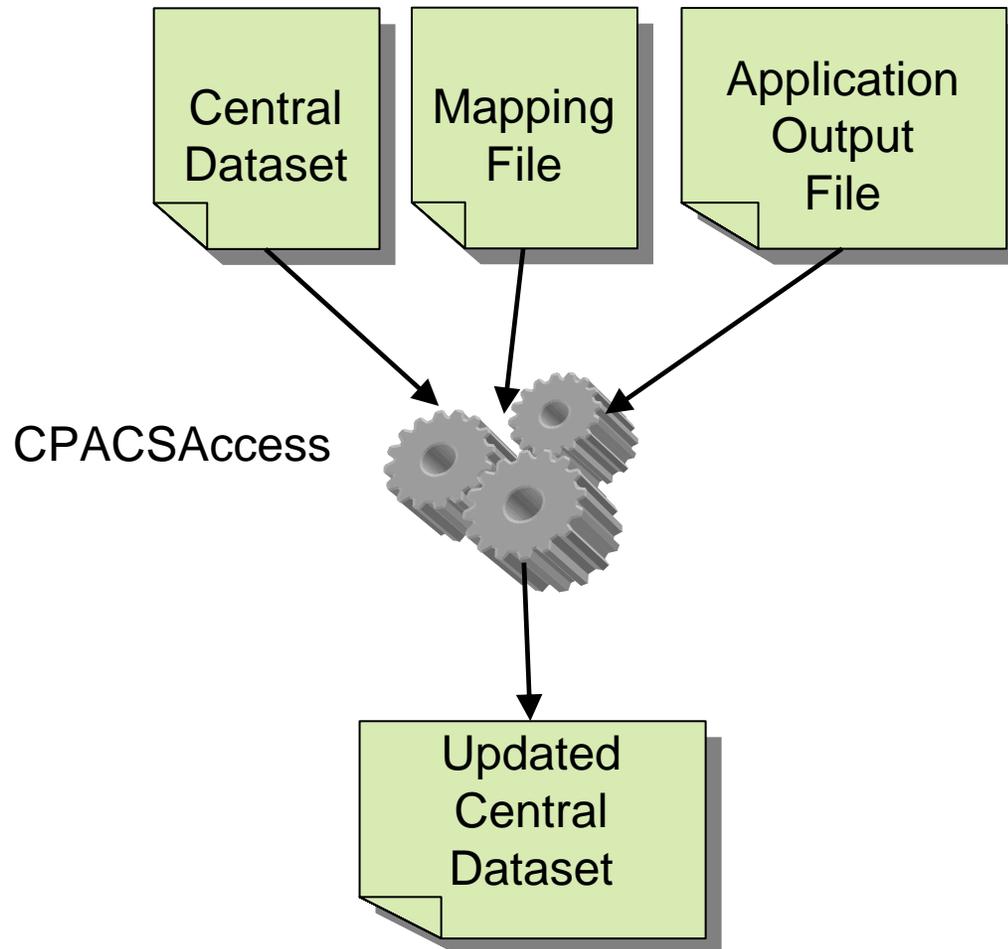
# Structure of the Central Dataset



# Data Export from the Central Dataset



# Data Import into the Central Dataset



# Example of a Mapping File

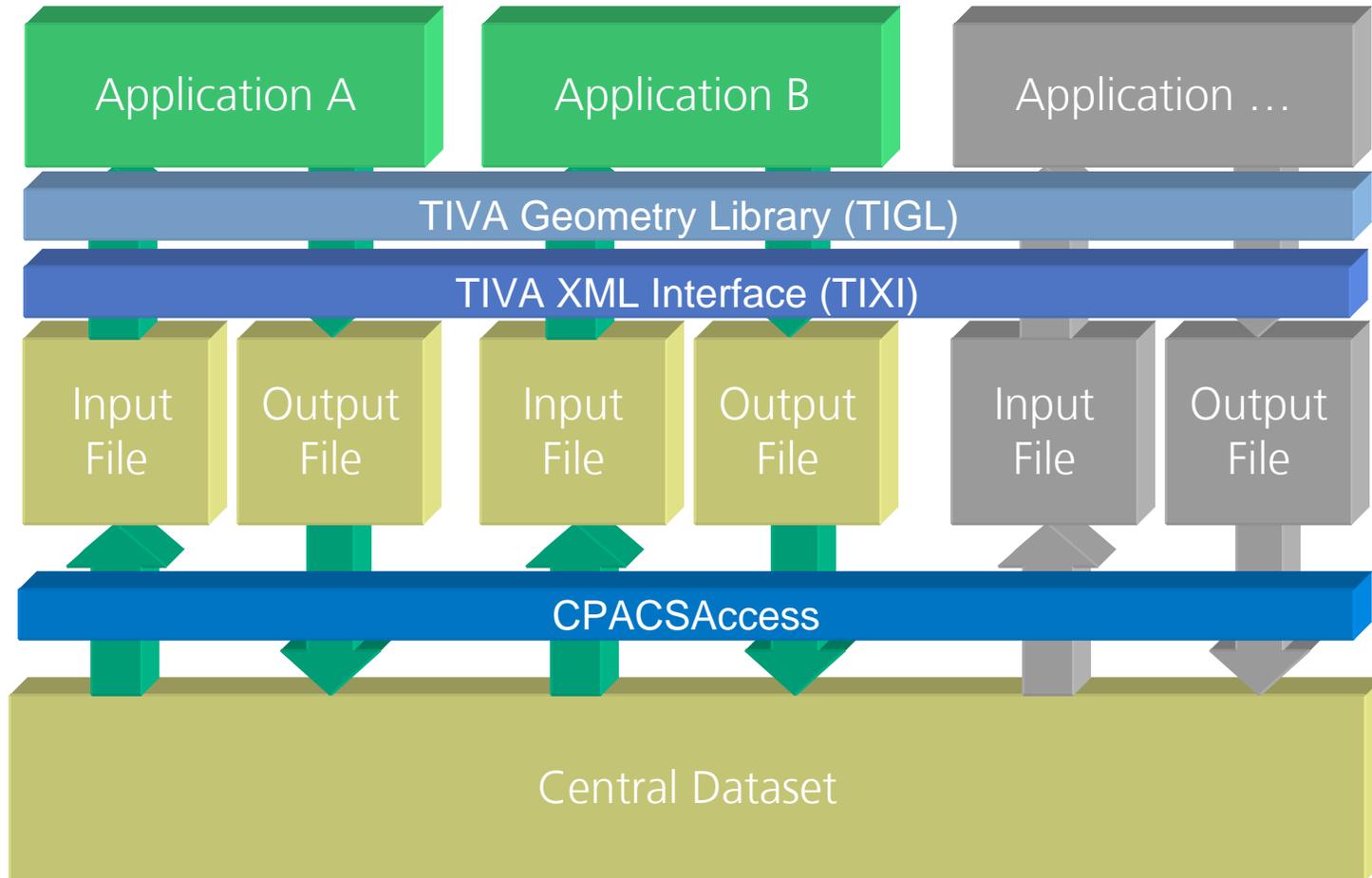
```
<?xml version="1.0" encoding="UTF-8"?>
<map:mappings xmlns:map="http://www.dlr.de/sistec/tool/mapping">

  <map:mapping>
    <map:source>/result</map:source>
    <map:target>/configuration/application[@name="IBUCK"]/result</map:target>
  </map:mapping>

  <map:mapping>
    <map:source>/result/values</map:source>
    <map:target>/configuration/common/values</map:target>
  </map:mapping>

  <map:mapping>
    <map:source>/result/old_name</map:source>
    <map:target>/configuration/common/new_name</map:target>
  </map:mapping>
  .
  .
  .
</map:mappings>
```

# TIXI – TIVA XML Interface (I)

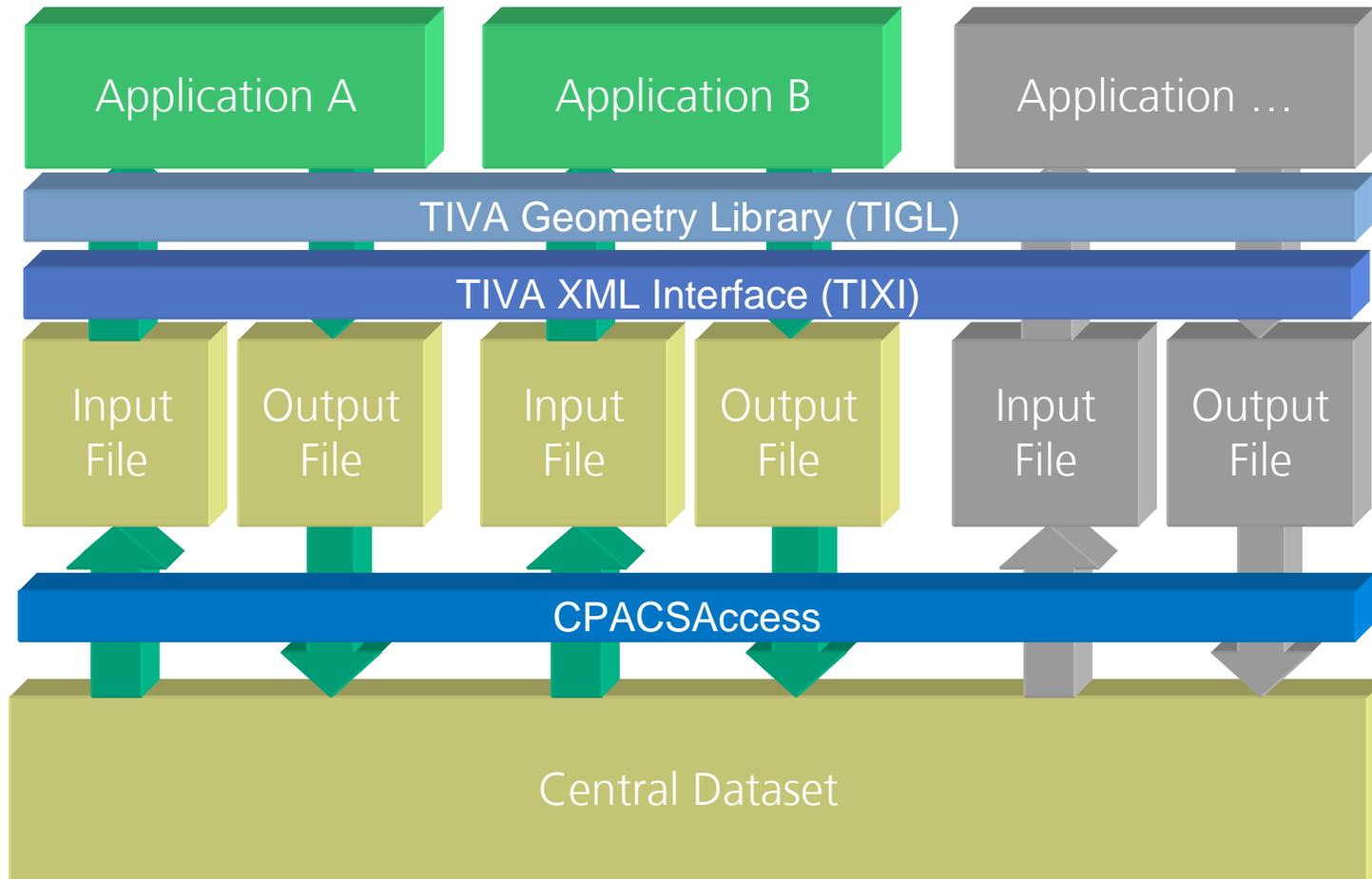


# TIXI – TIVA XML Interface (II)

- Library for XML I/O
- Uses libxml2 of the Gnome project
- Provides simple access to XML content through XPath expressions
  - Functions for reading and writing of
    - Strings
    - Floating point and integer numbers
    - Matrixes
    - 3D-Points
  - Checks for existence of elements
- C, Fortran, and Python interface

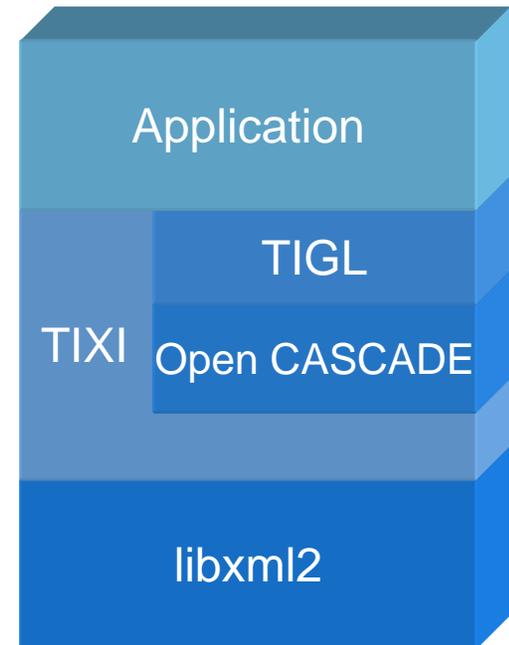


# TIGL – TIVA Geometry Library (I)

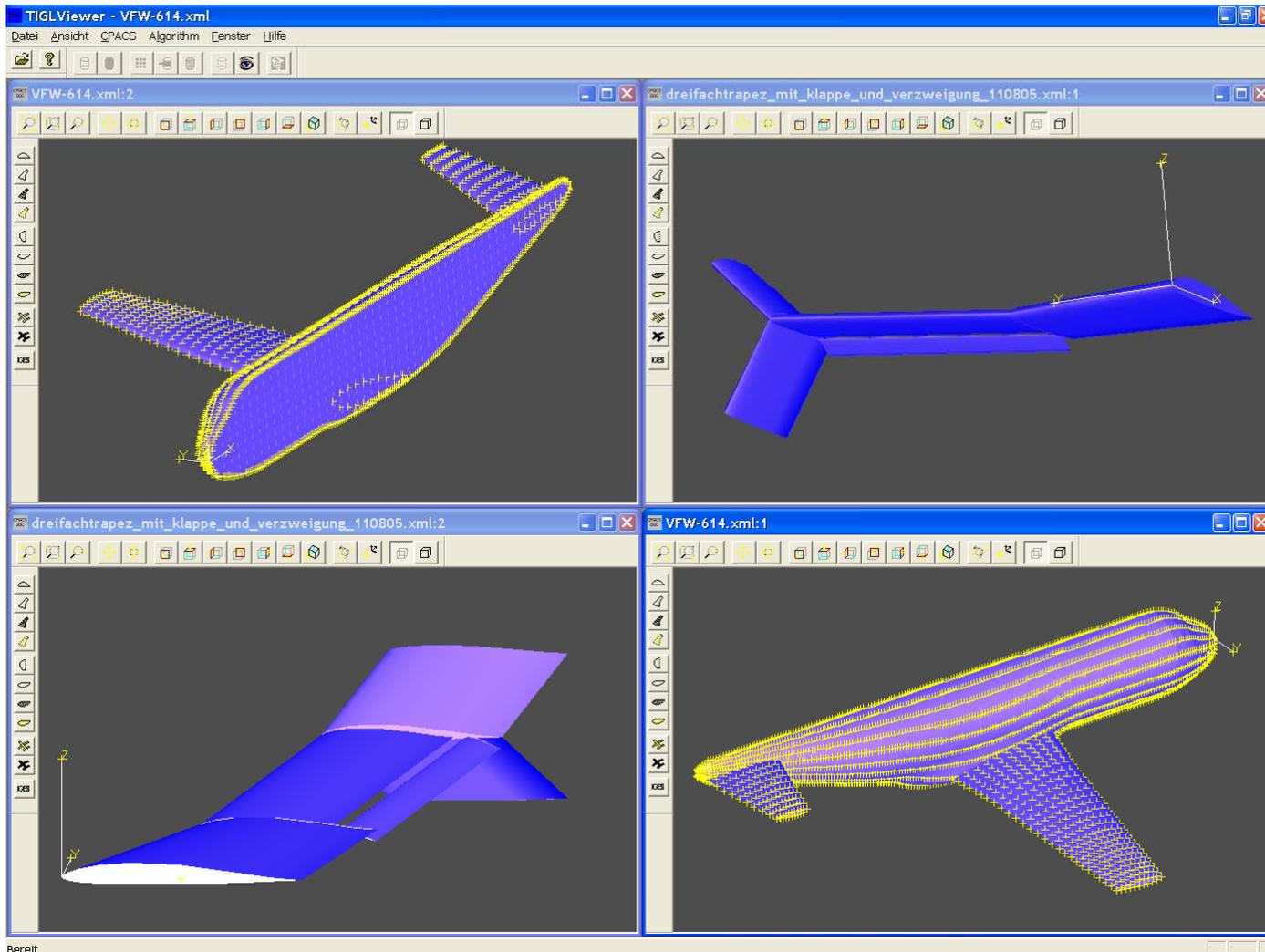


# TIGL – TIVA Geometry Library (II)

- Reading and processing the geometry information stored in the central dataset
  - Currently only for fuselages and wings
  - Uses Open CASCADE
- Construction of the 3D geometry
  - Creation of surfaces from cross sections
  - Used e.g. for calculation of surface points in absolute Cartesian coordinates
- Export of the airplane geometry in IGES or STL format
- C, Fortran, and Python interface



# TIGLViewer – Visualization Tool for TIGL



# Framework Integration (I)

- Integration framework used: ModelCenter
- Central dataset and applications components realized as ModelCenter plugins
  - Central dataset component
    - Interface between ModelCenter and central dataset
    - Update of the central dataset
    - Export and import from and into ModelCenter
  - Application wrapper component
    - Generic component that wraps an individual (standalone) application for use in ModelCenter
    - Generation of input files, application startup and mapping of results into central dataset
  - Controller component
    - Coordinates the components of the process chain

# Framework Integration (II)

The screenshot displays the Phoenix Integration ModelCenter 7.1.2 interface. The main window is titled "Phoenix Integration ModelCenter 7.1.2 - [C:\Modelcenter\Workshop\Projekte\Step\_4.pxc\*] - [Model (Analysis View)]". The interface is divided into several sections:

- Top Panel:** Contains a menu bar (File, Edit, View, Tools, Component, Project, Window, Help) and a toolbar with various icons for file operations and simulation.
- Left Panel (Model Tree):** A hierarchical tree view showing the structure of the model. The selected component is "LiftingLine\_with\_TIGL". The tree includes:
  - Positioning (dihedralangle: 4.39, section\_1: 1, section\_2: 2)
  - Positioning\_2, Positioning\_3
  - Name: VFW-614...
  - Wing\_2
  - Fuselages
  - Airfoils
  - Fuselage
    - Transformation (scaling, translation, rotation)
    - Sections
    - Segments
    - Positionings (name: VFW-614 f...)
  - Polars
  - Weights
  - Comment
  - Loads
  - ConfigurationZIPFromDataStore
  - CPACSFFromDataStore
  - CPACSToDataStore
  - LiftingLine\_with\_TIGL (selected)
  - BoxBeam\_with\_TIGL
  - Controller
- Right Panel (Diagram):** A component diagram showing the integration of the selected component. It features:
  - CPACS:** A component with a red phoenix icon, connected to the Controller.
  - Controller:** A central component with a green lightbulb icon, connected to CPACS, LiftingLine\_with\_TIGL, and BoxBeam\_with\_TIGL.
  - LiftingLine\_with\_TIGL:** A component with a blue TIGL icon, connected to the Controller.
  - BoxBeam\_with\_TIGL:** A component with a blue TIGL icon, connected to the Controller.
- Bottom Panel:** A workspace area containing:
  - A "favorites" section with a tree view of common, Drivers, Primitives, component plug-in, solution archive, and aserv://localhost.
  - Three icons representing loaded components: Controller.pxc, CPACS, and de.dlr.modelcenter.se...

# Summary and Outlook

- Steps to set-up a framework for collaborative engineering:
  - Define a common data format
  - Enable applications to use it
  - Integrate separate tools into a workflow system
  
- Future work
  - Implementation of an interface to STEP
  - Extension of the central dataset:
    - Geometry modeling for other construction units
    - Mission control data
  - Application of tools and common data format in other projects