jPAD — A Java Toolchain of Computer Programs for Aircraft Design.

Software Engineering Best Practices Applied to Aerospace Sciences

A. De Marco

jPAD — Java Programs for Aircraft Design

- A software toolchain for aircraft preliminary design and MDO.
- A modern, user friendly, modular framework.
- Support for simultaneous management/analysis of several aircraft and/or ‘varied’ configurations of the same aircraft.
- Conceived for collaborative design activities.
- Interoperability with other tools/disciplines (CAD/CFD/FEM analysis).
- Started in early 2014. Approx. 120k lines-of-code.
- Planning to go open source.
What you can do with jPAD

- Define parametric representations of wings, fuselages and nacelles with XML configuration/input files (similar to OpenVSP).
- Generate CAD geometries of aircraft assembly and sub-components (Open CASCADE, JNI Technology). Measure lengths, areas, volumes. Export in CAD formats (Brep, STEP, IGES, STL, Collada).
- Vary geometric parameters and regenerate internal representation of geometries programmatically.
- Import CPACS aircraft configurations files (Tigl Native Interface) and extract relevant properties. (Work in progress)
- Perform various types of analysis (L0, L0.5, L1): Aerodynamics, Stability & Control, Performance, Weight, Costs. (Structural TBD)
- Exports analysis results in XML (native/CPACS) and Excel formats.
We are aerospace engineers ... Should we extend our capabilities?
Software engineering principles

- **Software design patterns**: formalized best practices that the programmer can use to solve common problems when designing an application or system.

- Design patterns speed up the development process by providing tested, proven development paradigms.

- Examples of patterns:
  - Builder, Factory, Dependency Injection,
  - Strategy, Separation-of-Concerns,
  - Observer (Publish/Subscribe)

- Micro patterns (**design decisions in code**): DRY (Don’t Repeat Yourself), Sampler (Controlled Creation),
Example of Java code

```java
Aircraft aircraft = new Aircraft("Baseline_AC_AGILE.xml");
OperatingPoint op = OperatingPointFactory.getPoint("OP_001.xml");
ACAnalysisManager analysis = new ACAnalysisManager(op, aircraft,
    AnalysisTypeEnum.AERODYNAMIC,
    AnalysisTypeEnum.BALANCE,
    AnalysisTypeEnum.WEIGHTS,
    AnalysisTypeEnum.PERFORMANCE,
    AnalysisTypeEnum.COSTS);

analysis.calculateGeometryAuxiliaryData();
analysis.runAnalysis(AnalysisTypeEnum.AERODYNAMIC,
    AnalysisTypeEnum.PERFORMANCE);

DataWriter dataWriter = new DataWriter(analysis);
dataWriter.exportToXML("Analysis_Baseline_AGILE_op_001.xml");
dataWriter.exportToXLS("Analysis_Baseline_AGILE_op_001.xlsx");

CADBuilder cadBuilder = new CADBuilder(aircraft);
cadBuilder.build().exportToSTEP("Baseline_AGILE_op_001.stp");

CPACSBuilder cpacsBuilder = new CPACSBuilder(aircraft); // TBD
cpacsBuilder.build().export("Baseline_AGILE_op_001.xml");

WriteUtils.serializeObject(aircraft, aircraft.getName());
```
Java. Why?

- “Compile once. Run it everywhere.” (well, almost)
- Widely supported, continuously updated and improved.
- Many open source libraries available, especially for I/O tasks and for complex mathematical operations.
- Widely supported GUI frameworks (SWT/JFace and JavaFX) and a GUI visual builders.
- Object-Oriented paradigm is naturally applied in the abstraction of typical Aircraft Design problems.
- Promotes modularity: easier to work with in an ever changing team.
Parametric Fuselage in jPAD
Parametric wing in jPAD
Output files

- XML
- Microsoft Excel
- Charts
- CAD model ready to be meshed
Example of jPAD output

External fuselage shape exported as STEP file
Example of jPAD output

<table>
<thead>
<tr>
<th>Description</th>
<th>Unit</th>
<th>AIR72</th>
<th>F100</th>
<th>A320</th>
<th>B747_100B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material density</td>
<td>kg/m³</td>
<td>2711.0000</td>
<td>2711.0000</td>
<td>2711.0000</td>
<td>2711.0000</td>
</tr>
<tr>
<td>Maximum zero fuel mass</td>
<td>kg</td>
<td>19998.2451</td>
<td>34774.75497</td>
<td>56206.95596</td>
<td>192272.98603</td>
</tr>
<tr>
<td>Maximum landing mass</td>
<td>kg</td>
<td>20698.42095</td>
<td>40968.19348</td>
<td>68104.61636</td>
<td>305739.45542</td>
</tr>
<tr>
<td>Maximum take off mass</td>
<td>kg</td>
<td>22998.24550</td>
<td>45516.48164</td>
<td>75668.59596</td>
<td>339684.42269</td>
</tr>
<tr>
<td>Fuselage mass</td>
<td>kg</td>
<td>3203.3333</td>
<td>5378.0000</td>
<td>8392.6667</td>
<td>32349.6667</td>
</tr>
<tr>
<td>Wing mass</td>
<td>kg</td>
<td>2332.5000</td>
<td>4722.0000</td>
<td>8519.0000</td>
<td>40195.0000</td>
</tr>
<tr>
<td>HTail mass</td>
<td>kg</td>
<td>212.0000</td>
<td>461.5000</td>
<td>619.5000</td>
<td>3429.5000</td>
</tr>
<tr>
<td>VTail mass</td>
<td>kg</td>
<td>275.0000</td>
<td>369.5000</td>
<td>459.0000</td>
<td>1938.3333</td>
</tr>
<tr>
<td>Nacelle mass</td>
<td>kg</td>
<td>143.5000</td>
<td>389.5000</td>
<td>753.0000</td>
<td>1321.5000</td>
</tr>
<tr>
<td>Landing gear mass</td>
<td>kg</td>
<td>763.0000</td>
<td>1656.3333</td>
<td>2076.0000</td>
<td>13485.5000</td>
</tr>
<tr>
<td>Structure mass</td>
<td>kg</td>
<td>7072.8333</td>
<td>13366.3333</td>
<td>21572.1667</td>
<td>96684.0000</td>
</tr>
<tr>
<td>Power plant mass</td>
<td>kg</td>
<td>1329.14176</td>
<td>4134.85560</td>
<td>7647.87160</td>
<td>22087.08000</td>
</tr>
<tr>
<td>Systems mass</td>
<td>kg</td>
<td>2324.25622</td>
<td>3231.22575</td>
<td>4447.76585</td>
<td>12935.29415</td>
</tr>
<tr>
<td>Furnishings and Equipment mass</td>
<td>kg</td>
<td>1252.00000</td>
<td>1853.00000</td>
<td>2891.00000</td>
<td>0</td>
</tr>
<tr>
<td>Manufacturer empty mass</td>
<td>kg</td>
<td>11978.23131</td>
<td>22585.41468</td>
<td>36558.80412</td>
<td>131706.37415</td>
</tr>
<tr>
<td>Crew mass</td>
<td>kg</td>
<td>306.05819</td>
<td>459.08729</td>
<td>535.60184</td>
<td>1377.26187</td>
</tr>
<tr>
<td>Operating Items mass</td>
<td>kg</td>
<td>585.95600</td>
<td>939.25300</td>
<td>1292.55000</td>
<td>4739.35000</td>
</tr>
<tr>
<td>Operating empty mass</td>
<td>kg</td>
<td>12870.24551</td>
<td>23983.75497</td>
<td>38386.95596</td>
<td>137822.98603</td>
</tr>
<tr>
<td>Passengers mass</td>
<td>kg</td>
<td>6732.00000</td>
<td>10791.00000</td>
<td>14850.00000</td>
<td>54450.00000</td>
</tr>
<tr>
<td>ZeroFuelMass</td>
<td>kg</td>
<td>19602.24551</td>
<td>34774.75497</td>
<td>53236.95596</td>
<td>192272.98603</td>
</tr>
</tbody>
</table>
ADOpT: a GUI for jPAD

Example of project tree

The GUI – Eclipse SWT/Jface technology, JavaFX for the 3D view
ADOpT: a GUI for jPAD

Varying geometric parameters
ADOpT: a GUI for jPAD

CAD generated from parametric representation
ADOpT: a GUI for jPAD

Configuring the aerodynamic analysis
Interface with external CFD tools
Key Points

- Java programming language (JDK ≥1.8)
- Eclipse IDE
- Great effort to make code easily maintainable
- In-house solutions for:
  - Importing/exporting data (including CPACS via Tigl interface)
  - Array handling
  - GUI design (SWT/JFace libraries, JavaFX)
  - Building the CAD model (via Open CASCADE libraries/Java Native Interface JNI)
  - CAD 3D viewer (JavaFX)
  - DATCOM-like database for Aerodynamics/Performances/Stability & Control analysis modules (Hierarchical Data Format, HDF)
Conclusions

- Main features and general arrangement complete
- Work in progress:
  - Loads
  - Stability & Control module
  - Collaborative features:
    - CPACS
    - United States Air Force Stability and Control Digital DATCOM
    - FlightGear simulator input XML file