Hamburg, 11 November 2022

Invitation to the CPACS developer meeting

Dear CPACS developers,

Next week **Thursday, November 17**, there will be our next regular developer meeting.

When: 09:30 – 12:00
Where: Online (Webex)

We would like to propose the following agenda:

1. Schema development
   a. Provenance in CPACS: Revision of the *updates* node
   b. Weight and Balance: Status of the discussions
   c. Systems definition: Status and how to proceed with larger changes
   d. Status CPACS 3.5
   e. Slot for spontaneous topics

2. Library development
   a. Overview of current work on TiGL

If you wish to add anything to the agenda, just let us know until Tuesday, November 15 (cpacs@dlr.de). Below you find some material to prepare for the discussions.

Provenance in CPACS: Revision of the updates node

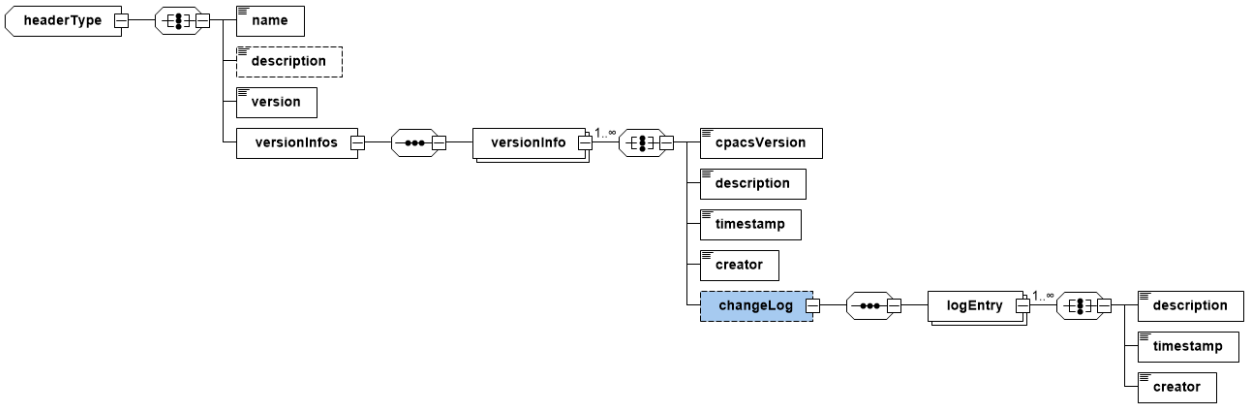
GitHub: #796 #797

Background:

In the CPACS header we have the so-called updates element. Here, information is stored about who changed what in the dataset and when.

The problem is that there are inconsistencies and ambiguities with the current definition. For example, it is not entirely clear where the version number is taken from: from the version element in the header node or from the last update node? Does an initial dataset immediately get an update node directly, even if it is not yet an update as such? In which order are update nodes kept? Etc...

In the last developer meeting a small working group was organized for this, consisting of Carsten, Jan, Anton and Marko. We came up with the following revision:



The main idea is that the version element also serves as a link to the corresponding versionInfo node, which carries this version as an attribute (similar to uID linking; implemented via xsd:key and xsd:keyref). Here all further provenance information can be found, independent of the order of the nodes.

Target for developer meeting:

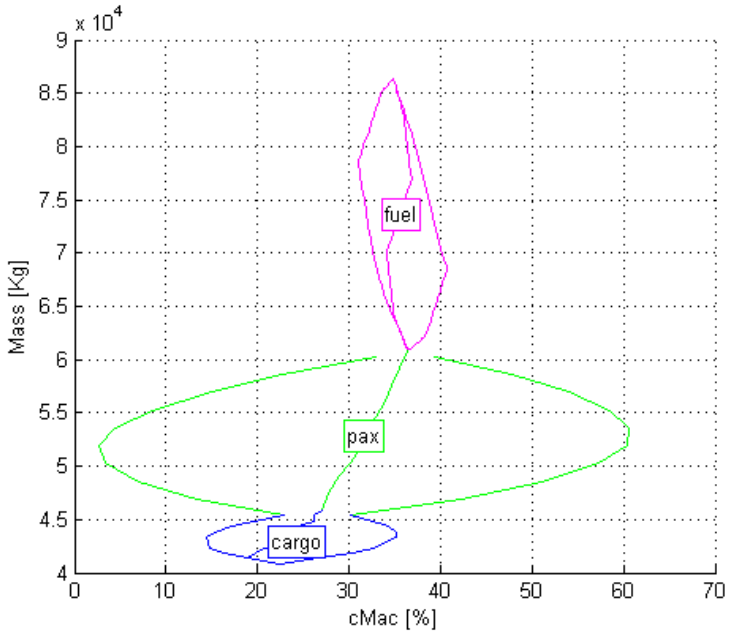
This is a breaking change. We could implement this in the next minor release (3.5). However, TiGL would need a patch as well, in order to import the cpacsVersion from the new location. **This TiGL update would be required to read CPACS 3.5 then. Can we accept this circumstance?**

Weight and Balance: Status of the discussion

GitHub: #636, #724, #725

Background:

The weightAndBalance element provides vectors describing various trajectories of mass vs. coG (center of gravity):



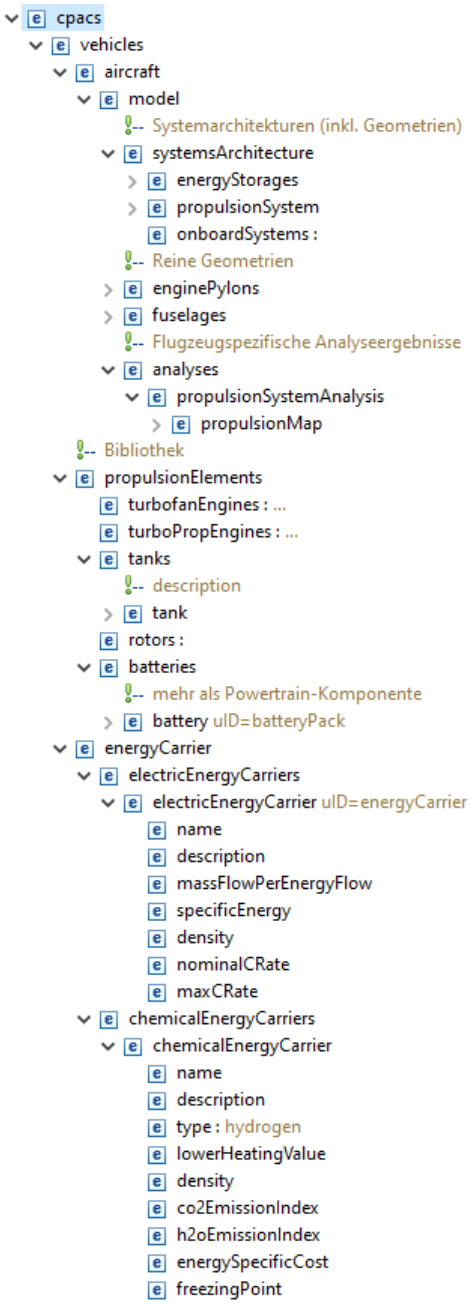
The problem is that high-fidelity flight mechanical analyses require information about configurational changes, which are triggered at certain conditions in the mission definition. How do we define that a rescue helicopter picks up a patient during the mission, how does this influence the center of gravity depending on where the rescue stretcher can be mounted in the cabin? How do we define when to use which effectors of a military aircraft along a mission, how do we determine the respective center of gravity shift?

Target for developer meeting:

We will provide **insight into the ongoing discussions**. Currently this is organized by Richard Kuchar as the main stakeholder (DLR - Institute of System Dynamics and Control).

Systems definition: Status and how to proceed with larger changes

GitHub: #606, #721, #732



Background:

Currently, it is hardly possible to define novel types of propulsion architectures, such as hybrid electric propulsion or propulsion technologies based on hydrogen.

Several problematic aspects come together:

- The engine element is a mix of classic turbofan and turboprop.
- The engine performanceMap is often used aircraft-specific, because there is no aircraft-specific performanceMap for the whole propulsion system.
- Propellers/Fans are modeled as little helicopters (rotorcraft) attached to aircraft wings...
- The fuel element is not general enough for electric flight.
- Missing hydrogen tanks
- Missing power breakdowns
- Missing definition of system architecture (components and their connections)

Target for developer meeting:

The developer meeting is **not intended to go deep into this issue** or even come up with solutions. A working group has already developed first data structures, which will be further elaborated and presented to a larger group in the next weeks.

The question is rather **how we deal with bigger changes** (e.g., renaming of engines) and how we manage the transition. Should a new major release (CPACS 4.0) be introduced for something like this? Or do we let old and new nodes run in parallel and somehow mark them as deprecated to gather more practical experience? Or other solutions?

Status CPACS 3.5

GitHub: CPACS 3.5 project board

Ideally, CPACS 3.5 will introduce an improved system description with many related topics.

Target for developer meeting:

Check out the planned issues for CPACS 3.5 on GitHub. Which ones should be increased in importance? **Who can take responsibilities?** Which ones might have become unimportant, etc. ...