# Alternate Proposal for Allowing User-Defined Data

**Here is a simpler alternate proposal for an extension to SIDS, to allow the user to put data (of arbitrary dimension) into the CGNS file. This is for arbitrary-sized user-defined stuff that needs to be put SOMEWHERE, but is currently not allowed by the API.**

**Under each of the following nodes:**

- **CGNSBase_t**
- **Zone_t**
- **GridCoordinates_t**
- **FlowSolution_t**
- **DiscreteData_t**
- **ConvergenceHistory_t**
- **ArbitraryGridMotion_t <-- note new location**

**allow the following child node:**

| | |
|---|---|
| **Name:** | **User defined** |
| **Label:** | **DataArray_t** |
| **Data-Type:** | **user defined** |
| **Dimensions:** | **user defined** |
| **Dimension Values:** | **user defined** |
| **Data:** | **user defined** |
| **Cardinality:** | **0,N** |
| **Parameters:** | **DataType, dimension of data, size of data** |
| **Child Nodes:** | **Figure 26** |

## Advantages to this new proposal:

- **This construct is identical to the one currently allowed under RigidGridMotion, ZoneIterativeData, and BaseIterativeData. So there is consistency.**
- **There are NO new API calls required! The user would access this data with cg_goto and cg_array_write or cg_array_read! The only thing that has to be done is to modify the API to ALLOW this child node to exist under the given nodes. Therefore, this change seems relatively simple to do.**
- **This construct easily gives a huge amount of flexibility to the user, for allowing code-specific data to be put in the CGNS file (which is what we desparately need).**
- **Naturally, adding this construct requires some changes to the documentation (SIDS and File-Mapping). But it does NOT require that an entire new section or chapter be written... rather, only existing sections need to be slightly modified.**

## Disadvantages:

**The full implications of allowing this under each of the above-mentioned nodes are not known. For example, currently under ConvergenceHistory_t, there are allowed an arbitrary number of DataArray_t's already, but they must all be of size (number_of_iterations). Would it be too confusing to have SOME arrays that are of arbitrary dimension, and SOME of dimension (number_of_iterations)?**

**It is definitely do-able (and there is no confusion) to allow it under:**

- **CGNSBase_t**
- **Zone_t**

**and under ArbitraryGridMotion_t and GridCoordinates_t there are only short lists of DataArray_t data-name identifiers that require a certain dimension. So allowing additional arbitrary DataArray_t's shouldn't be too difficult here (it would be like what is currently done under RigidridMotion, for example).**

**But under FlowSolution_t and ConvergenceHistory_t, a very large list of DataArray_t data-name identifiers exist, that require a certain dimension. And under DiscreteData_t, the list is infinite. Thus it could be very difficult / confusing to allow additional DataArray_t nodes of ARBITRARY dimension here.**

**This begs the question: do we really NEED to allow arbitrary-sized DataArray_t nodes under ALL of the above-mentioned nodes, or is it enough to allow them only under CGNSBase_t, Zone_t, ArbitraryGridMotion_t, and GridCoordinates_t?**