FAILY, S. 2007. *Living with flight dynamics: proposals and possible pitfalls for harmonising flight dynamics systems with EGOS*. Presented at the 2nd European ground system architecture workshop (ESAW 2007), 12-13 June 2007, Darmstadt, Germany.

Living with flight dynamics: proposals and possible pitfalls for harmonising flight dynamics systems with EGOS.

FAILY, S.

2007

Living with Flight Dynamics : Proposals and Possible Pitfalls for Harmonising Flight Dynamics Systems with EGOS

Shamal Faily

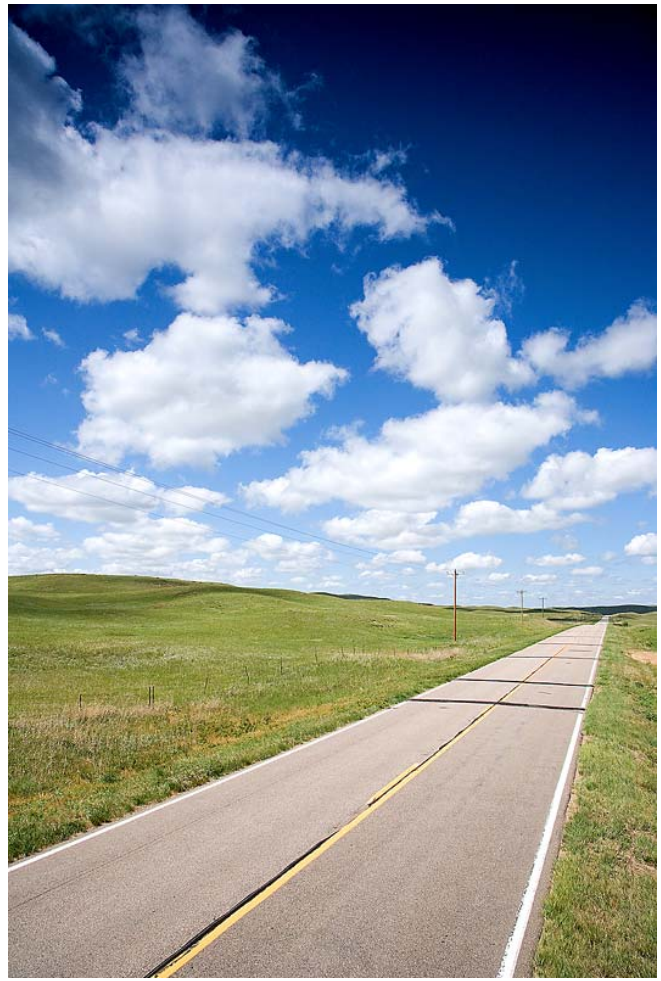
Consultant

A case for harmonisation

- Minimise maintenance by maximising re-use.
- Increased productivity = more time to deal with evolving Flight Dynamics user requirements.

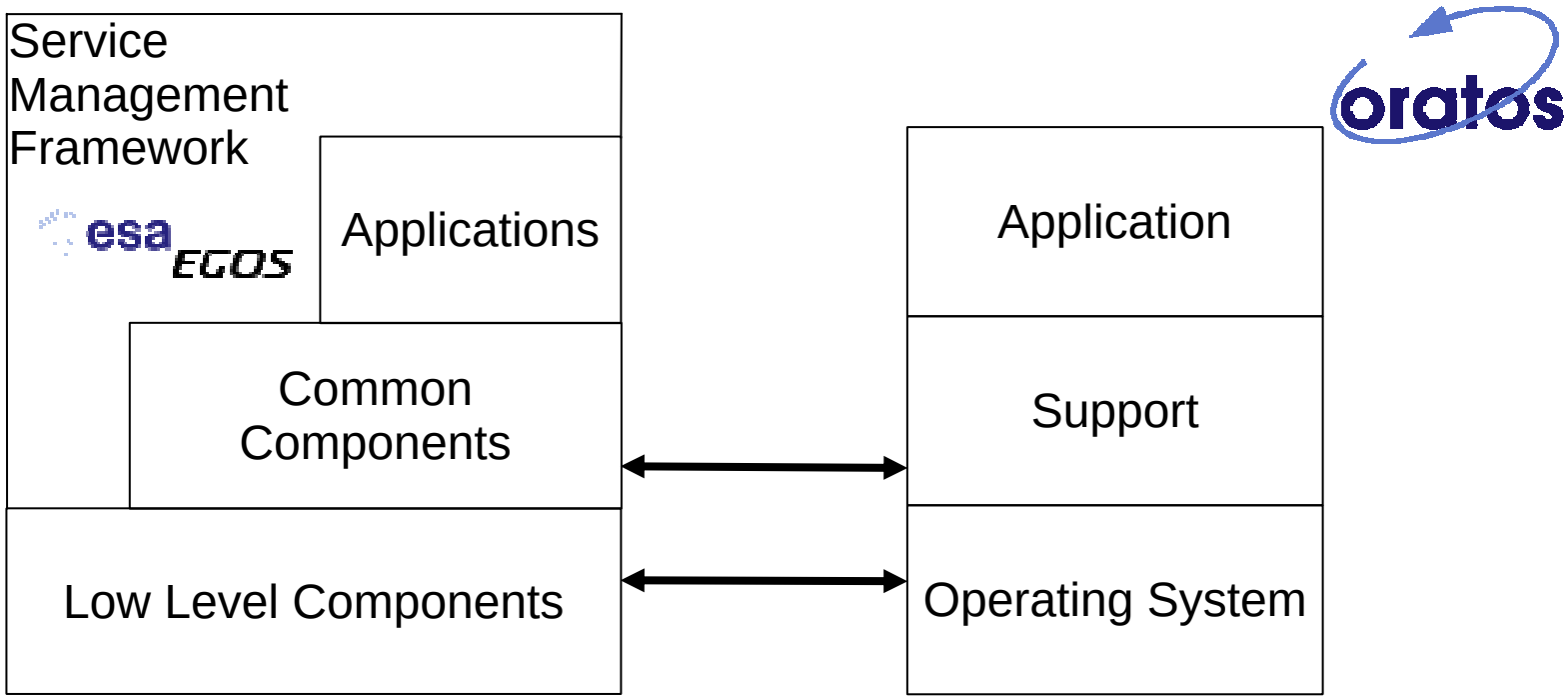
Agenda

- 4 proposals for harmonisation
  - Telemetry Monitoring Software
  - Low Level/Common Components
  - Open Source/COTS usage
  - Generic Software Requirements
- For each proposal
  - The case for harmonisation
  - The possible pitfalls



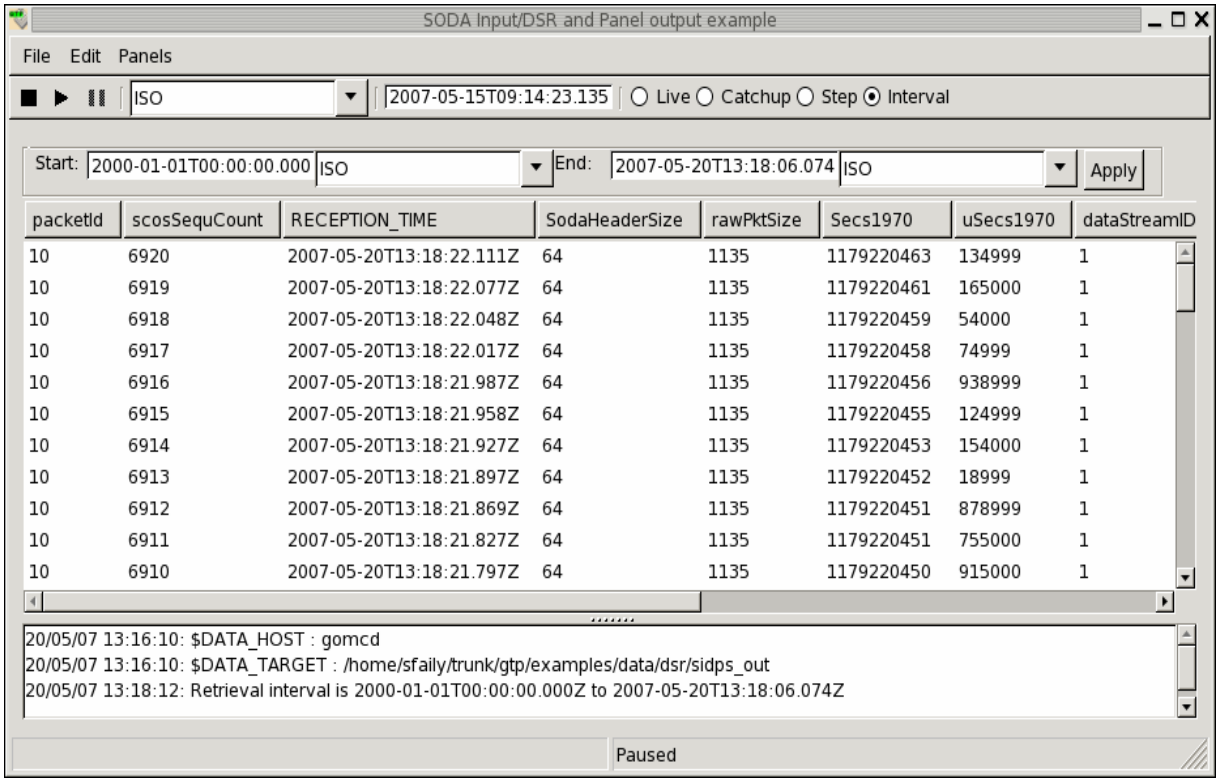
EGOS and ORATOS

- SCOS-2000/ORATOS commonality study indicated commonalities do exist.
- Both architectures have evolved since study, but commonalities should still hold.



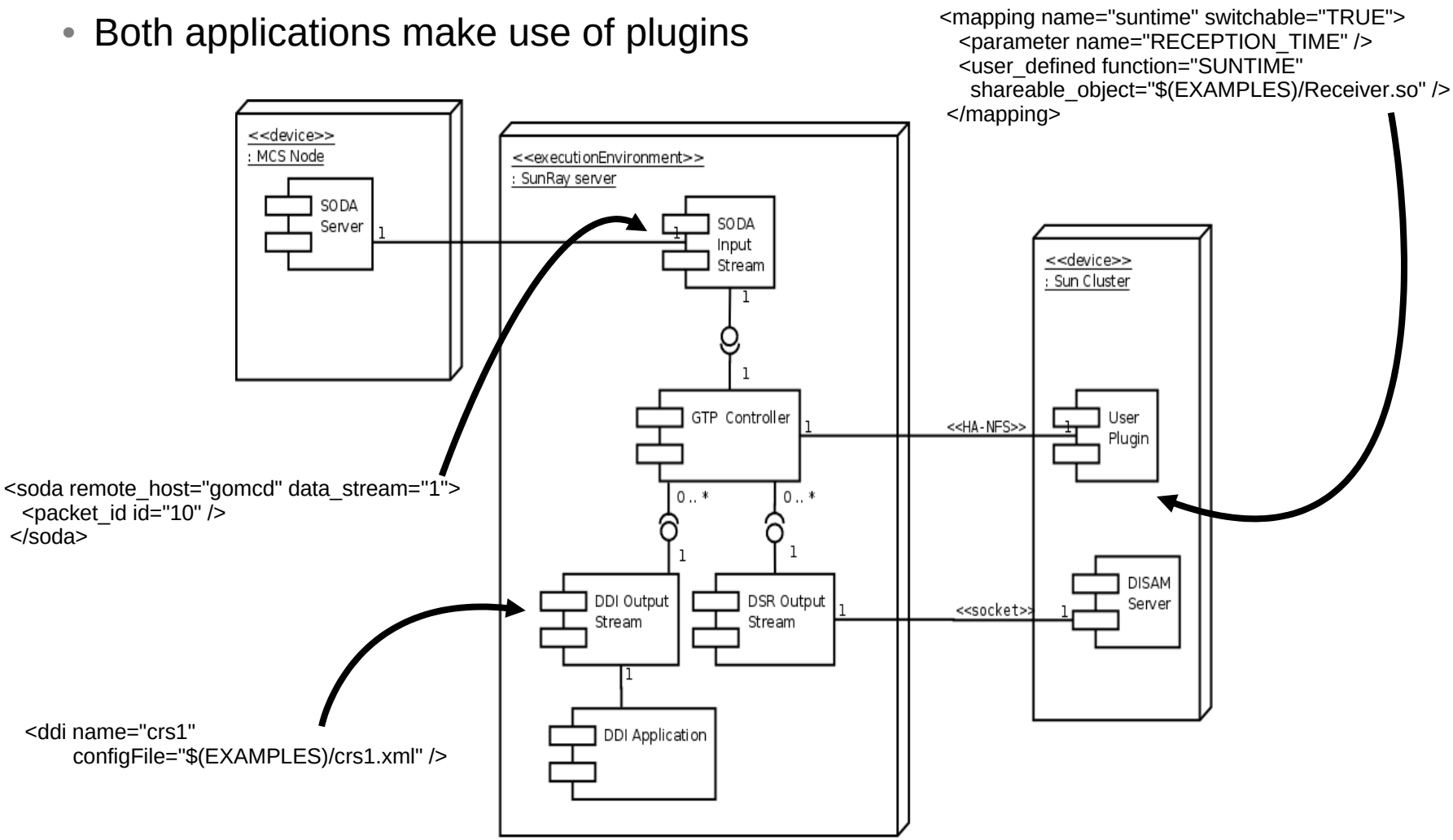
Telemetry Software Harmonisation : Proposal

- Analogies between EGOS (Telemetry Monitoring Desktop) and ORATOS (Generic Telemetry Processor)



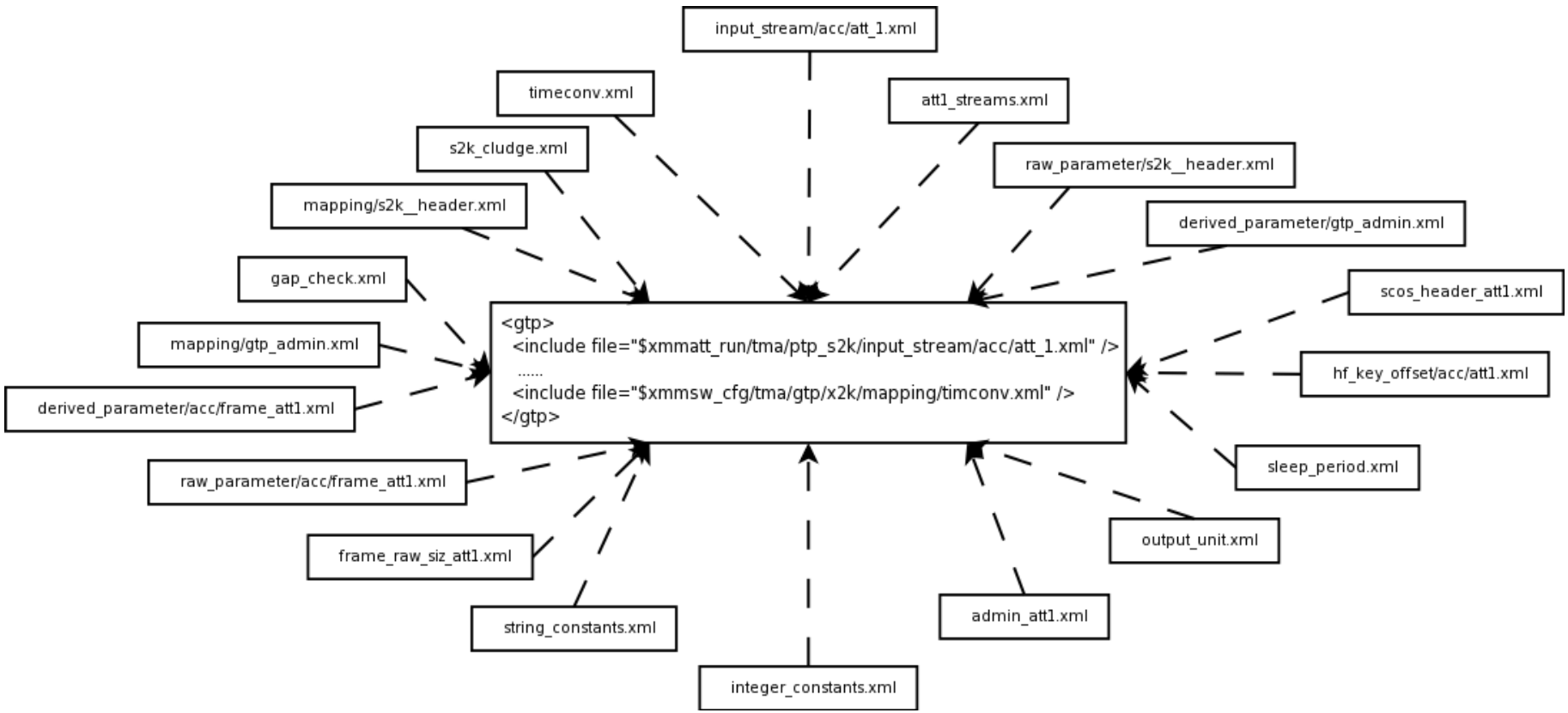
Telemetry Software Harmonisation : Proposal

- Both applications make use of plugins



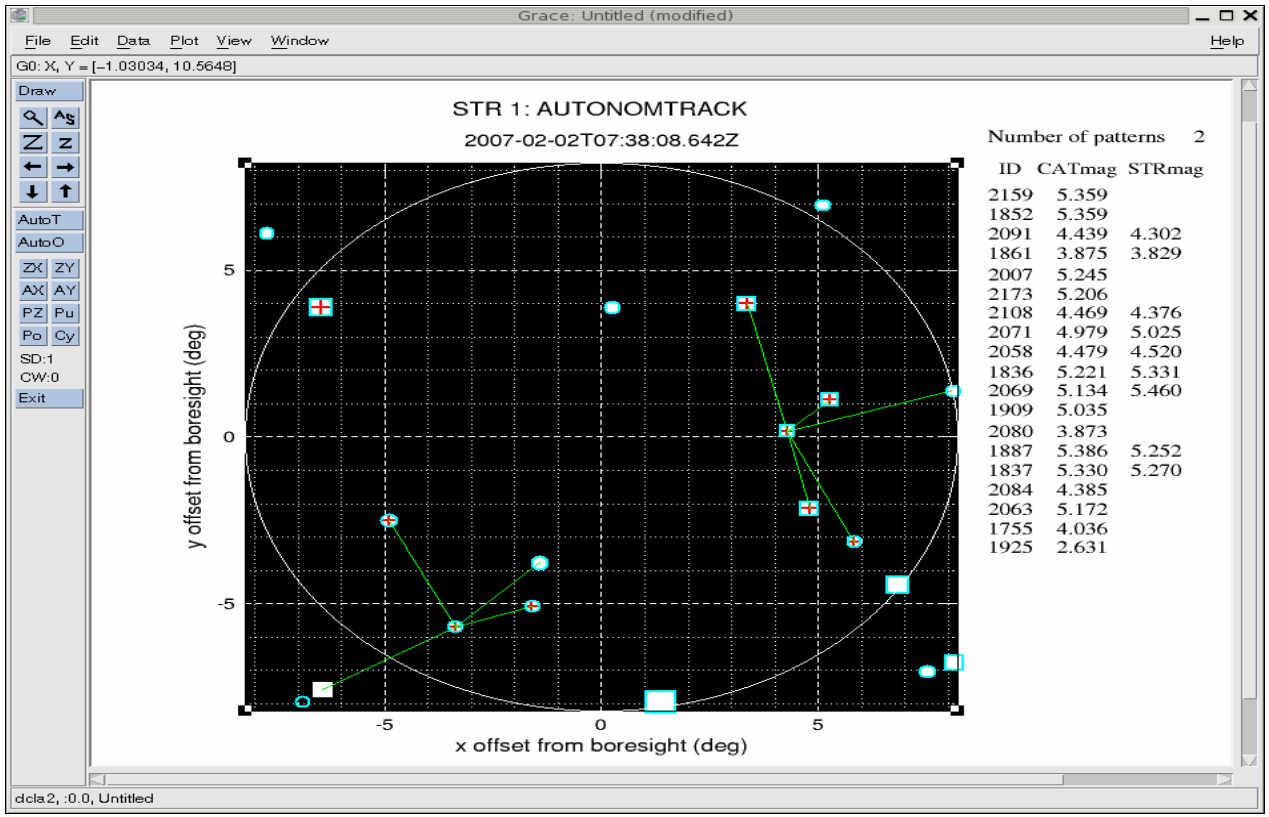
Telemetry Software Harmonisation : Proposal

- GTP Mission specific configuration can be non-trivial.



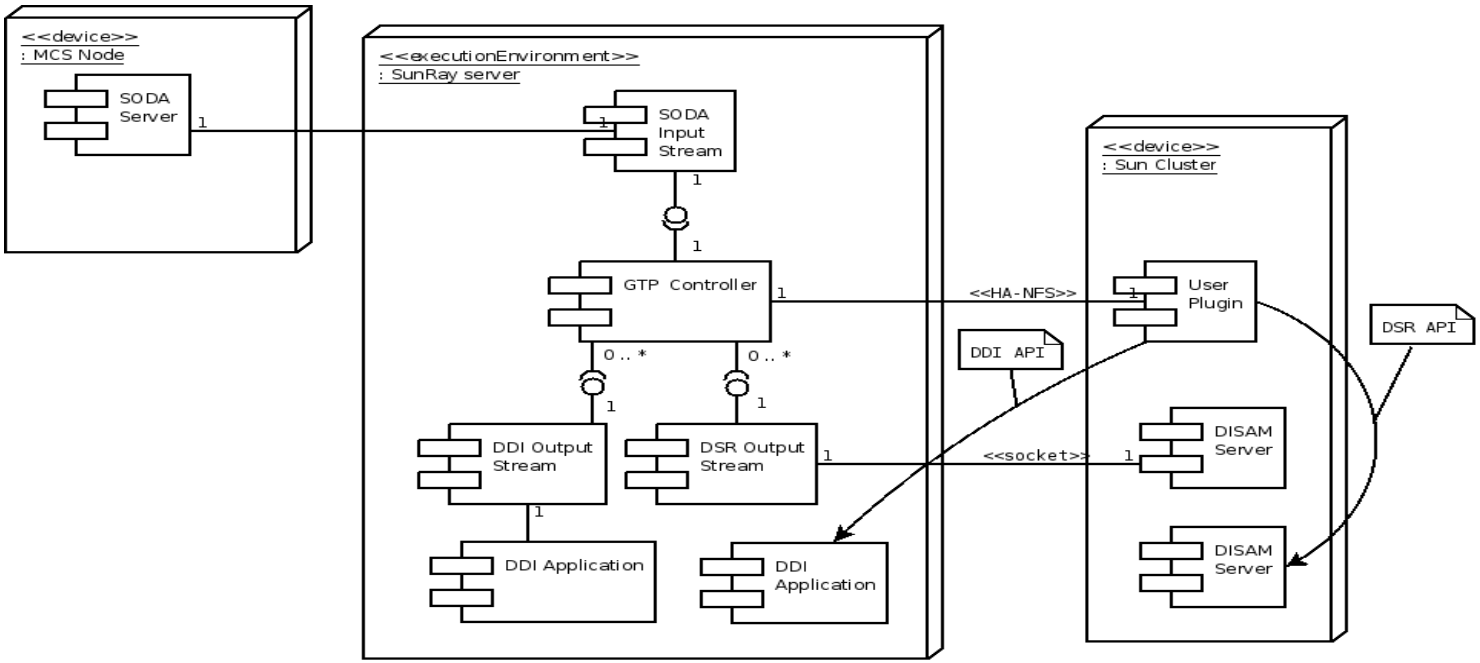
Telemetry Software Harmonisation : Proposal

- GTP Parameter visualisation uses “best-of-breed” open source.



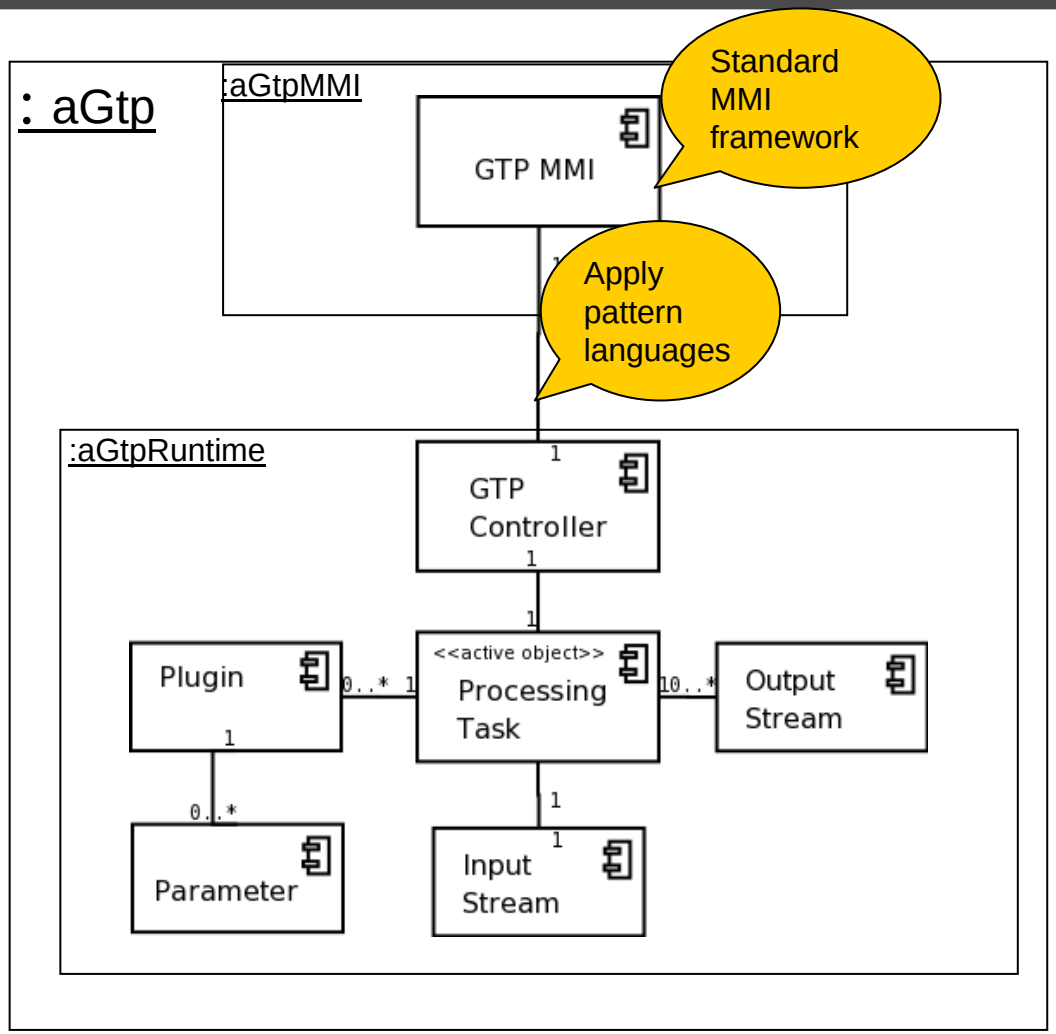
Telemetry Software Harmonisation : Possible Pitfalls

- Supporting End-User Development (EUD) is important!



- EUD experiences lead to new ORATOS requirements.

Telemetry Software Harmonisation : Possible Pitfalls

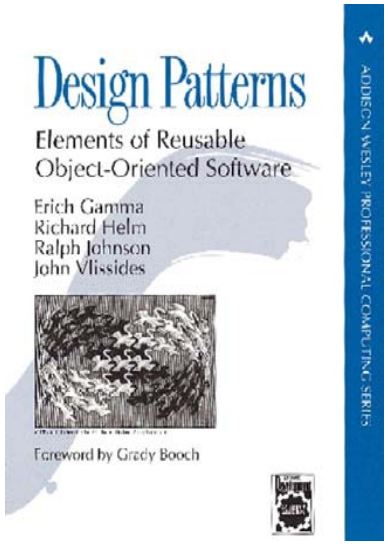


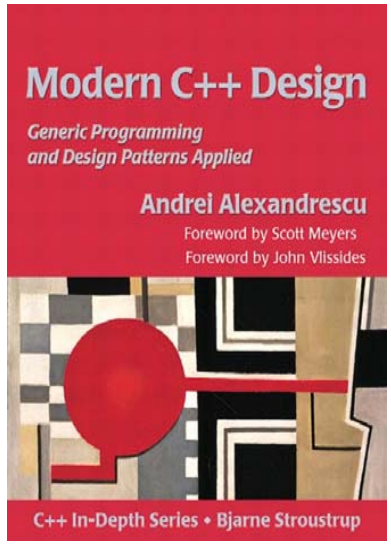
- What is the QoS impact of selections?
- What is the maintainability impact of selection given extra artificial complexity?
- Does it meet our operational requirements?

- Misunderstanding implicit assumptions can lead to unsustainable architectural mismatch.
- ..which may be unnecessary.

Low-Level / Common Components Harmonisation : Proposal

- Similarity between low-level and common component use and design principles.





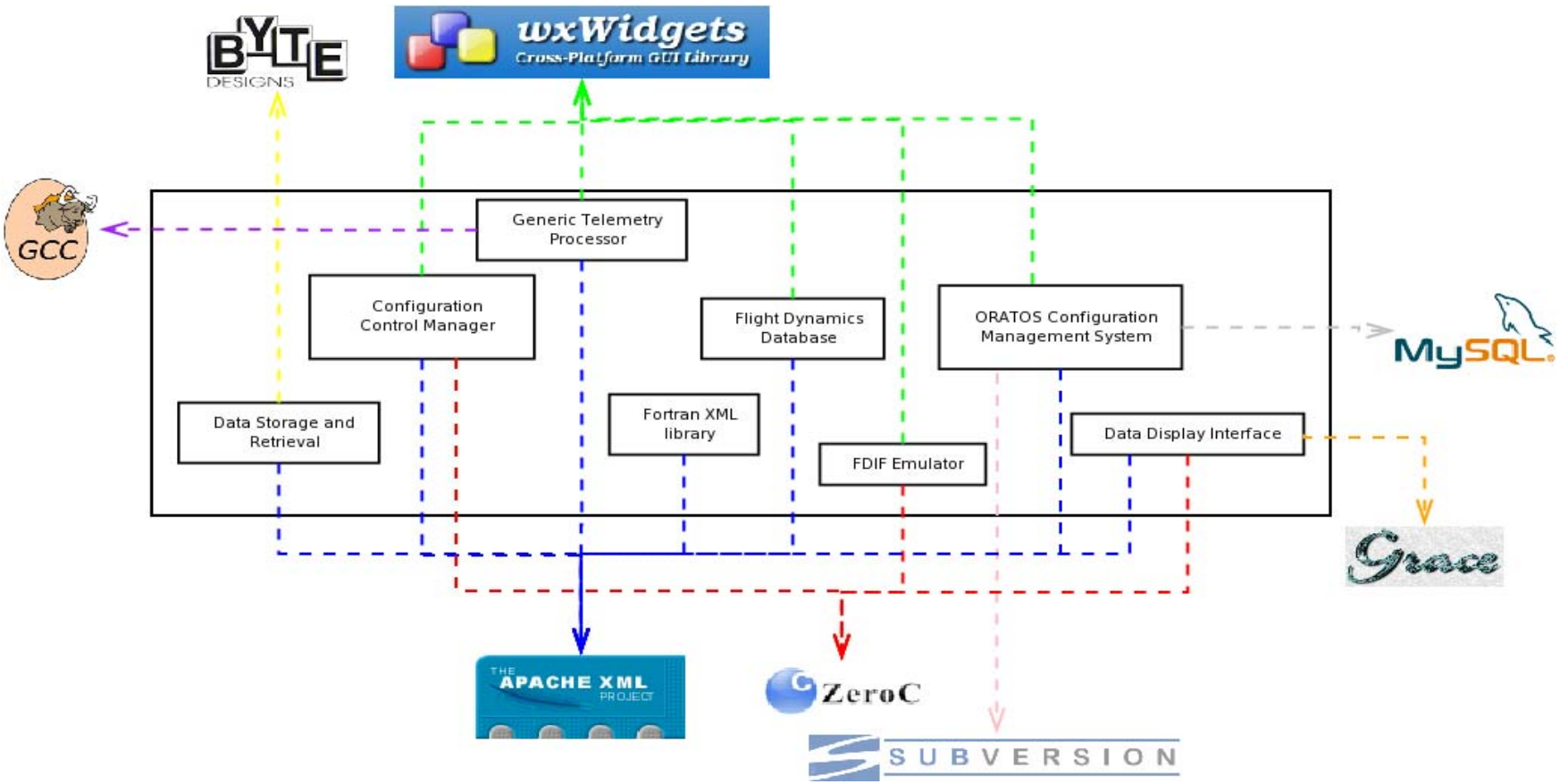
- Components developed for Flight Dynamics may be re-usable in other Algorithmic/Scientific applications.

Low-Level / Common Components Harmonisation : Possible Pitfalls

- ORATOS component re-use traditionally driven by tacit knowledge.
- Agreed interfaces need to be negotiated.
- What are the supportability requirements?

Open Source / COTS usage Harmonisation : Proposal

- ORATOS relies on Open-Source.



Open Source / COTS usage Harmonisation : Proposal

- Re-use of lessons learned replacing closed with open-source.
- Harmonisation of products and versions eases idiomatic software re-use.

Open Source / COTS usage Harmonisation : Possible Pitfalls

- Vendor supported “stable” versions not always stable when integrated with Flight Dynamics software.
- Flight Dynamics Open-Source selection based on experience developing and maintaining it as part of the Flight Dynamics infrastructure.
  - Subversion more robust than CVS when updating the repository structure.
  - ICE easier to use and components easier to maintain than CORBA (ACE/TAO).

Generic Software Requirements Harmonisation : Proposal

- Many EGOS Generic Software Requirements already implicitly met, e.g. Cluster awareness requirements.
- Harmonising means setting common standards for 3rd parties to develop components used in Flight Dynamics Systems.

Generic Software Requirements Harmonisation : Possible Pitfalls

- Current design and implementation constraints heavily biased towards C++ and Java.
- Mandating use of explicit CASE tools stifles the initiative of those who wish to use complementary tools and methods.

$$
\begin{array}{|l}
\textit{MappingInit0} \\
\hline
\textit{Mapping}' \\
\textit{mappingName}? : \textit{String} \\
\textit{routineName}? : \textit{String} \\
\textit{library}? : \textit{SharedLibrary} \\
\hline
\textit{routineName}? \in \operatorname{dom} \textit{library}?.\textit{routines} \\
\textit{name}' = \textit{mappingName}? \\
\textit{routineObject}' = \textit{library}?.\textit{routines}\ \textit{routineName}? \\
\hline
\end{array}
$$

$$\textit{MappingInit} \mathrel{\widehat{=}} \textit{MappingInit0} \lor \textit{RoutineNotPresent}$$

$$
\begin{array}{|l}
\textit{RoutineNotPresent} \\
\hline
\textit{routineName}? : \textit{String} \\
\textit{library}? : \textit{SharedLibrary} \\
\hline
\textit{routineName}? \notin \operatorname{dom} \textit{library}?.\textit{routines} \\
\hline
\end{array}
$$

Summary

- A foundation for harmonisation exists.
- All pitfalls can be overcome by requirements negotiation.
- Flight Dynamics users as stake-holders of EGOS should be the basis of subsequent requirements negotiation.

Thank you for listening!

- Any questions ?