Euclid

The challenge of Euclid SGS (Science Ground Segment) Architecture

C. Dabin (CNES)

http://www.euclid-ec.org
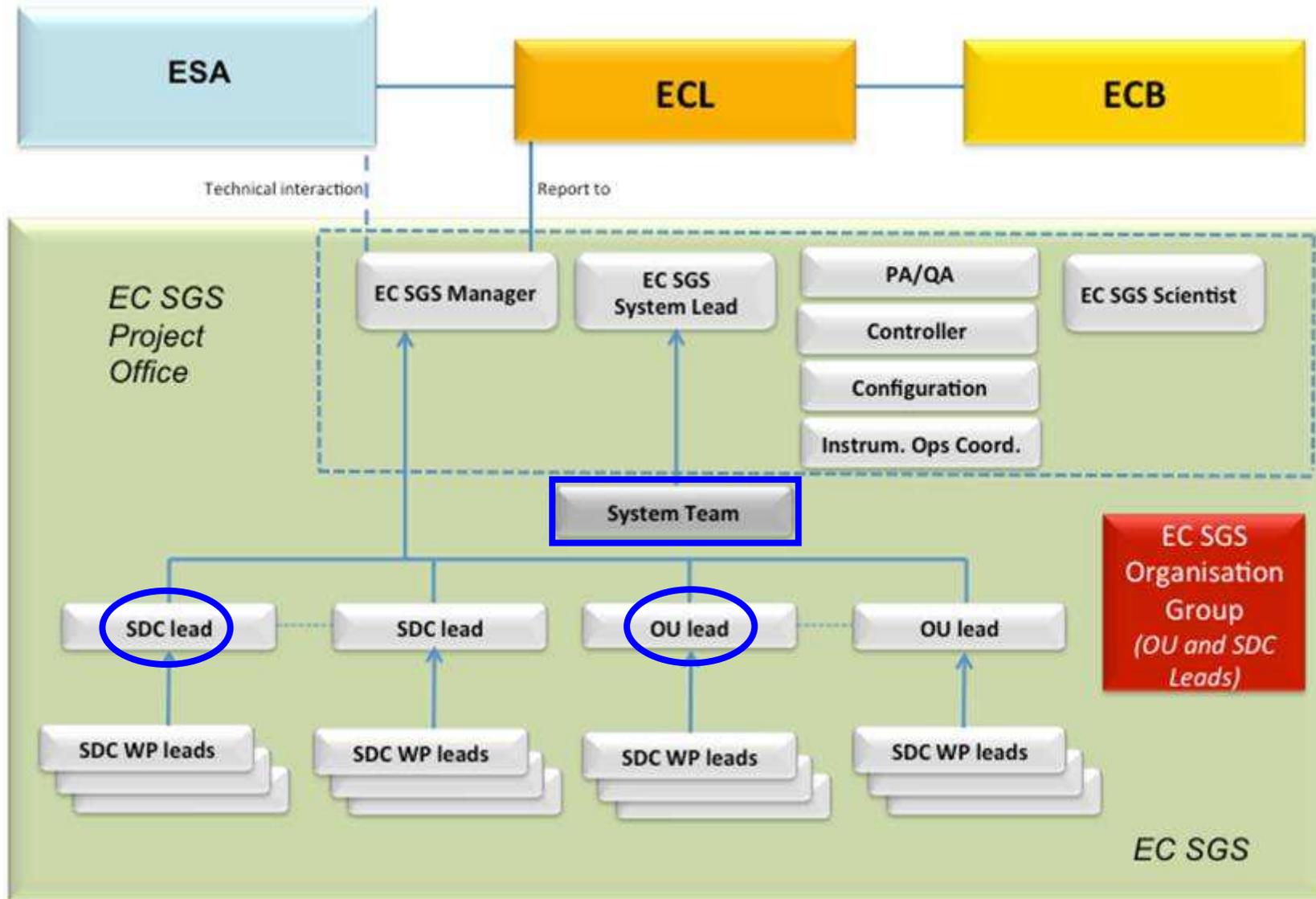
'Coordinate the overall activity aimed at defining standards and development procedures for the SGS : system architecture, common tools, APIs, tests, integration and validation, Euclid Archive System, Data distribution within the EC ' (from SIP)

- **System Team Coordination**
- **Architecture and performance**
- **Pipeline interface & Infrastructure Abstraction Layer**
- **Monitoring & Control**
- **Data Modelling**
- **Integration/validation/Qualification/Acceptance Tests (Support to V&V)**
- **Euclid Archive System**
- **Data transfer**
- **Data & Processing Orchestration**
- **Common Tools**
- **Common Data Quality Control Tools**
- **Level 1 data processing common infrastructure**

Key challenges

- Heavy **simulations** before the mission
    - Codes are running : TIPS, E2E code, VIS SIM !
    - Terabytes are requested
- heavy (re)**processing** needed from raw data to science products (e.g. on Planck volume multiplied by dozens)
- amount of **data** that the mission will generate per yearly release :
    - 26 PBytes of data => storage !
    - 1.10^{10} objects => database !
- **accuracy and quality** control required at every step

Data flows optimization is a key driver

- decrease as much as possible data access
- decrease as much as possible data exchange
- keep a very thin infrastructure between software modules and data access to avoid overheads while
- accessing data locally inside each SDC infrastructure
- allow the inclusion of new national SDCs, as needed
- simplify as much as possible the system design

Euclid Consortium

⇒ **Architecture infrastructure scenarii** under evaluation with respect to the 'Big picture' data flow required for Science

⇒ 'Traditional' Architecture

    ⇒ Infrastructure (SDC Production) dedicated for one or 2 OU's
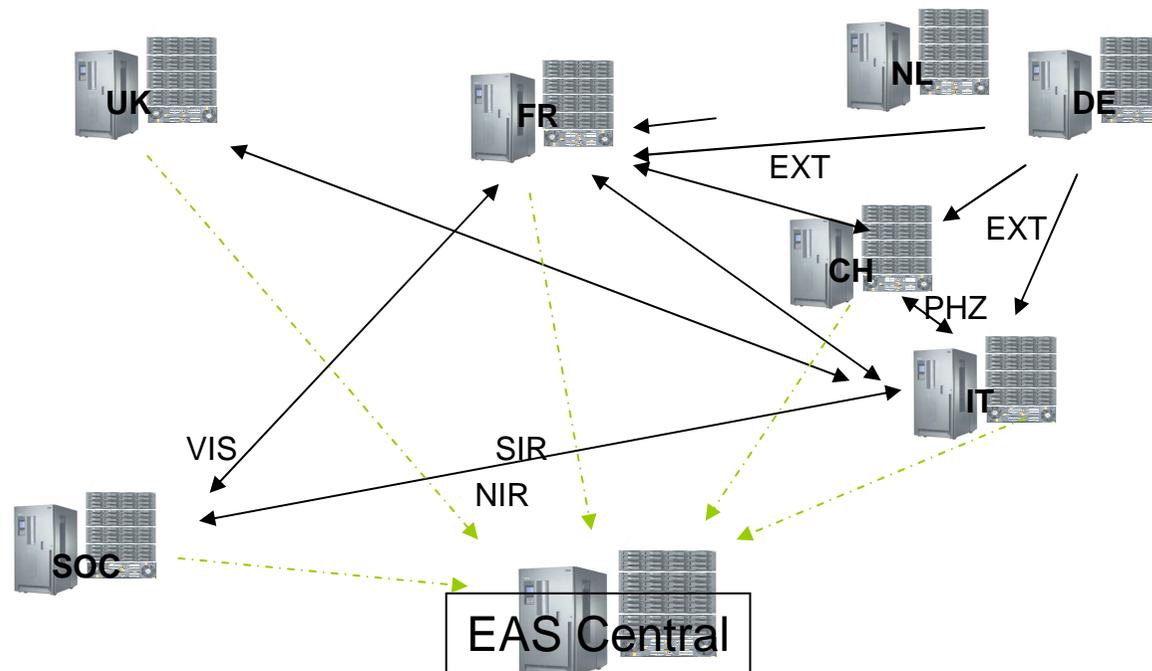
UK   FR   NL   DE

EXT

CH   EXT

PHZ

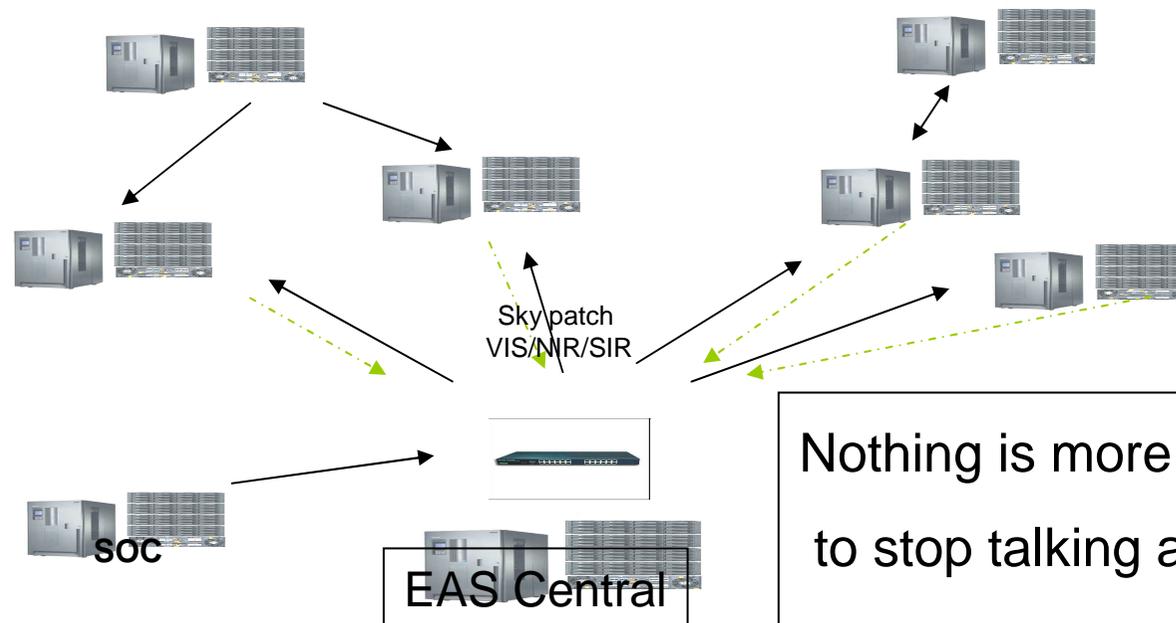IT

VIS   SIR

NIR

SOC

EAS Central

⇒ **Architecture infrastructure scenarii** under evaluation with respect to the 'Big picture' data flow required for Science

'Distributed' Architecture

⇒ Any OU on any SDC and pipeline aggregated

⇒ Sky splitting on infrastructures… as far as possible

Sky patch
VIS/NIR/SIR

soc

EAS Central

Nothing is more convincing than

to stop talking and start doing

So bench and mock up !

Pipeline interface & IAL

- Needs of minimal set of commonality

- Needs for **infrastructure abstraction** : IAL (Infra. Abstraction Layer), virtualization

- **Infrastructure Abstraction Layer** (IAL) isolates the pipeline from the underlying infrastructure, so makes integration easier and facilitates deployment on platforms
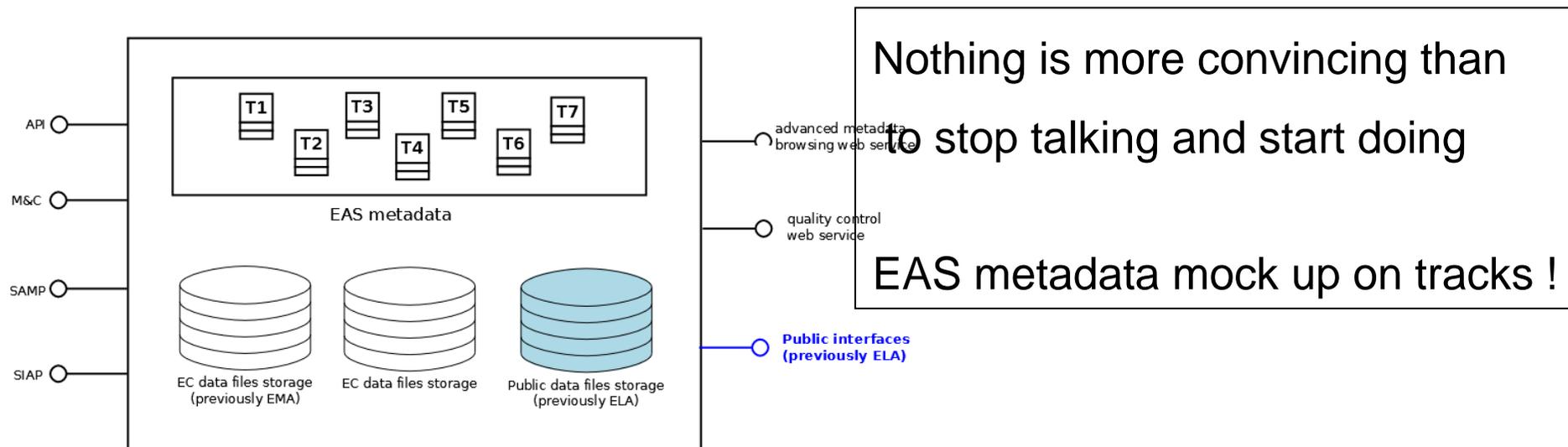
> Nothing is more convincing than
>
> to stop talking and start doing
>
> IAL mock up on the way !

- To avoid laxness in the description of exchanged data

- To get a single, unambiguous definition of exchanged data

- To harmonise practices, thereby simplifying the processing of the different project interfaces (to systematize the production of documents, in particular)

- To formalise both syntactic (e.g. a real value with a given precision) and semantic (e.g. coordinates expressed in a certain reference frame) description

- Every single data is defined in a single XML schema with one root element

- Each single file should follow Naming Rules

Normative referentiel … already available… monthly meetings

- Data Centric
  - Formerly EMA (discussion .. And agreement between consortium and ESAC)
  - Technical challenge : huge volume, huge queries,…
  - Political constraint for a centralized approach,
  - Merge processing requirements and legacy requirements in a single document

Nothing is more convincing than to stop talking and start doing

EAS metadata mock up on tracks !

- No "correct" choice

- "Good Enough"™

- Choose and stick with it

- Don't all have to make same choices...

  ...but some things must be common

RECOMMENDATIONS AND…
IMPLEMENTATION
ON CONTINUOUS INTEGRATION PLATFORM (CODEEN)

C++/PYTHON   ECLIPSE SUBVERSION xUNIT MAVEN NEXUS Jenkins
Jira Sonar Doxygen Redmine Adobe Connect

Conclusion

- Amener les partenaires à une vision commune, partagée
  - Interfaces pipeline
  - Data Model
  - Architecture
  - Métriques logicielles

- Partager des outils (CODEEN) pour anticiper l'intégration

- Contribuer aux règles normatives

- Ecueils
  - Le normatif est contraignant … mais nécessaire and 'Just enough'
  - Sortir du bac à sable très tôt (difficile !)
  - Manque de communication