Using semantic web technologies for aerospace industrial data migration

Migrating complex PLM data

Olivier Rey

Airbus Helicopters for CNRS MaDICS/RoD 2019

June 27 2019

Version 1.0

AIRBUS

Agenda

Warning

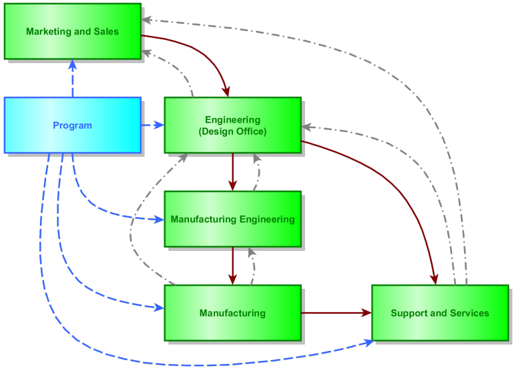
The architecture diagrams,
data diagrams and other technical information
in this presentation look like reality
but are not exact

Problem statement

Organization of an aerospace OEM

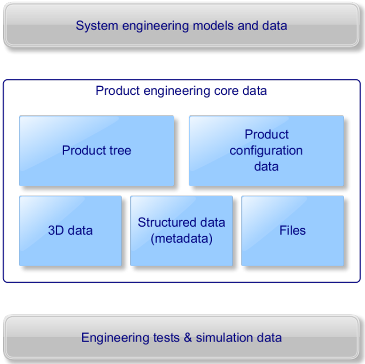
An aerospace OEM main operational actors are:

- The Program that defines the requirements for the A/C and pilots the budgets
- The Marketing and Sales that advertise and sell the A/C and its options
- The Engineering that designs the A/C with airworthiness constraints
- The Manufacturing Engineering that prepares the optimized production
- The Manufacturing that builds the A/C
- The Support and Services that supports the A/C and provide spare parts and services all along its lifecycle



The many interactions within an OEM

The engineering product data at the center of the company



Engineering core product data

The core product data are:

- Of various types (models, 3D, metadata, files, configuration data, simulation data)
- In various tools
- Not fully integrated together
- Consistent *but with great manual efforts*

The difficulty is that each of those data have a lifecycle and so, data integration is not trivial

Historically, the PLM systems were only dealing with product engineering core data

➪ *Today, every entity is digital and search for digital continuity*

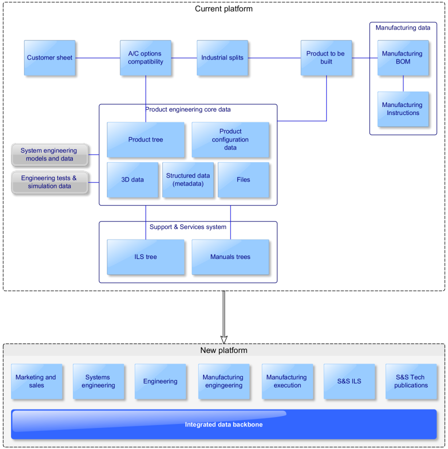
The move to an integrated platform

With years, the software vendors progressed and defined more and more integrated platforms that have the following characteristics:

- The platforms are proposing more typed data corresponding to the domain business objects
- Many different actors can store their data inside the platform
- Each data type has its own lifecycle
- Digital continuity is finally possible

Yes, but we have a complex data migration problem to solve:

- Input data must be made consistent (multiple sources)
- The source engineering data are “change-oriented”, while standard digital continuity requires engineering data to be “product-oriented”



The upcoming migration

Is it possible/reasonable to migrate?

The common practices, in the industry, is *to avoid migrating PLM data*, for the following reasons

- People are used to the current systems and training them on newer systems is expensive
- Newer systems are expensive and the migration business case is not obvious
- The IT migration procedures are not considered as reliable enough, so there are risks of data loss
- The track of changes, from an original certified aircraft to the aircrafts built today, could be lost and so the migration could jeopardize the type certificate holding

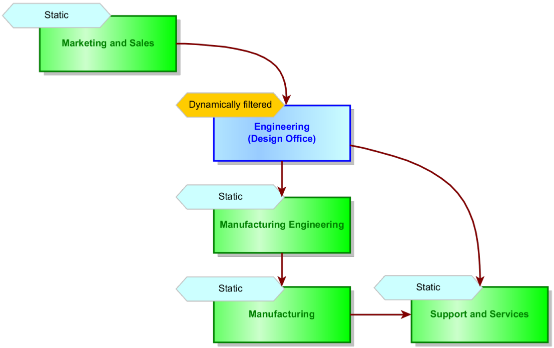
However

- Systems are getting very old (+20 years) and they are often running on old hardware and software
- They are complemented with many Excel spreadsheets, which generates many painful manual interventions, source of errors and inefficiency
- New generations of engineers would like to work in today's digital era
- Current systems make product families/product componentization hard to manage
- Collaboration with partners and suppliers is complex with current systems

➪ *With those old systems, product innovation is slower, more costly and error prone*
PLM data migration is one of the challenge of the digital transformation

The example of configuration data

Status on configuration data



Configuration management vision

The configuration data are at the center of the digital continuity ambition

However, configuration data are not homogeneous for all actors

- The engineering tools that are used today are based on a product that was designed to filter a so-called “150%” product tree
- This implies that the Engineering division is working based on *filters* to *see* a real instance of the aircraft
- As the result of the product filtering is not stored and has no lifecycle, interfaces to other systems are very complex to realize

Requirements for data conversion

There are two migration options:

- Migrating the data "as-is" and re-implement the specific filtering algorithms
- Go from migration to data *conversion*

Data migration is possible if we have a set of tools that enable this complex conversion

At the center of complexity is the fact that data have intricate lifecycles

- This point is important because it is the particularity of industrial systems
- Product lifecycle and change lifecycle are deeply connected

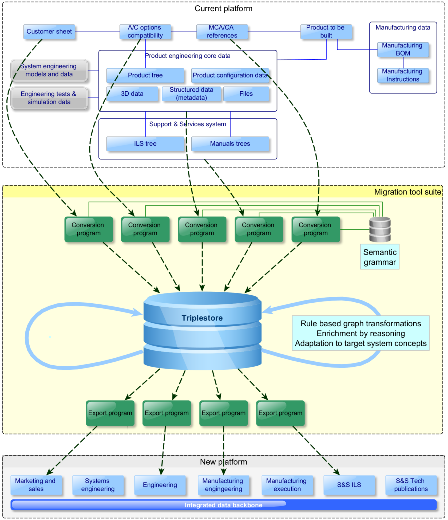
Requirements
Being able to aggregate several data sources and manage data consistency
Being able to manage "lifecycled" data
Being able to create a new semantic system from the current one (perspective change)
Being able to preserve the semantic history of objects (e.g. the change process and history s-tack)
Being able to build multiple views for multiple actors
Being able to adapt to the native concepts of the target system

Requirements for data conversion

Building a conversion framework based on semantic web technologies

The conversion framework is proposing

- Extractors from source systems
- Convertors that are taking as-is data and transform them into RDF data following a global semantic grammar (with a source URI)
  - ➪ All semantic links of original data are preserved in this operation
- A linked data repository containing the target semantic grammar
- Modification requests implementing rules (reasoning or enrichment rules) that will convert bijectively the source semantic graph into the target semantic graph (with a target URI)
- Extractors able to rebuild objects conforming to the target semantic grammar

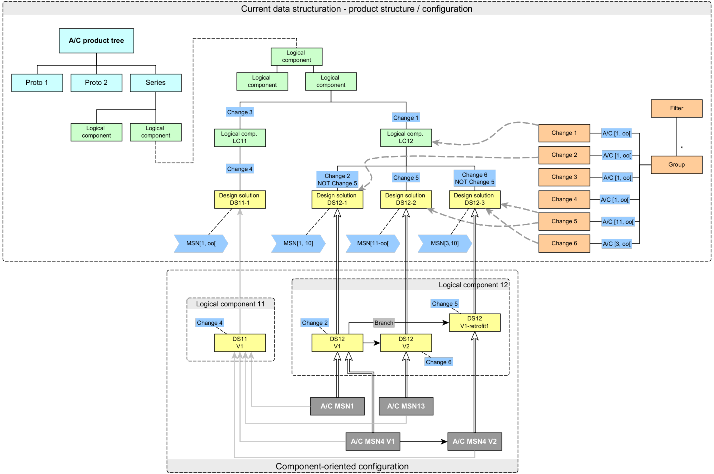


The step by step transformation

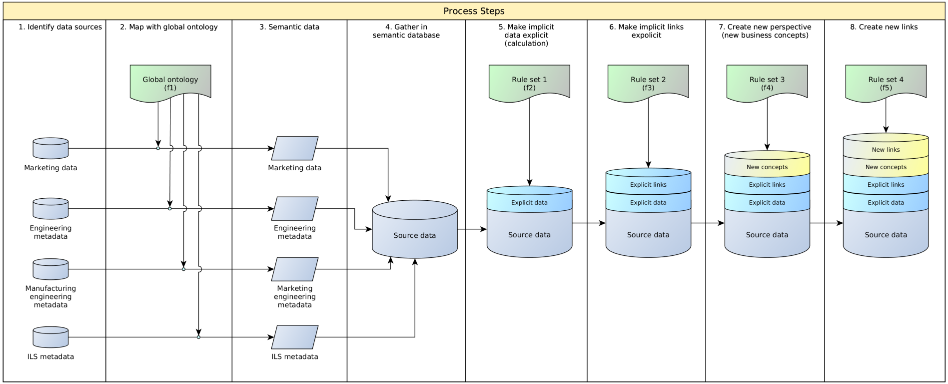
Reconstructing data on another semantic perspective

The objective is to convert data on another semantic perspective

- The new perspective is semantically equivalent but is enriched with new concepts
- It is "semantic rewiring" of the existing semantic graph
- The product and change lifecycles and history are preserved but they are packaged in new entities

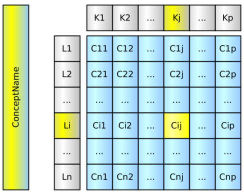


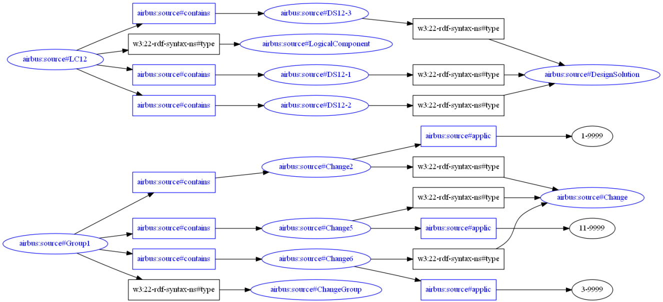
Transformation process overview



The transformation of data is done through a series of semantic graph transformations

Step 4: Import source data after RDF conversion





For each line, we can generate:

- `(type` L_i `ConceptName)`

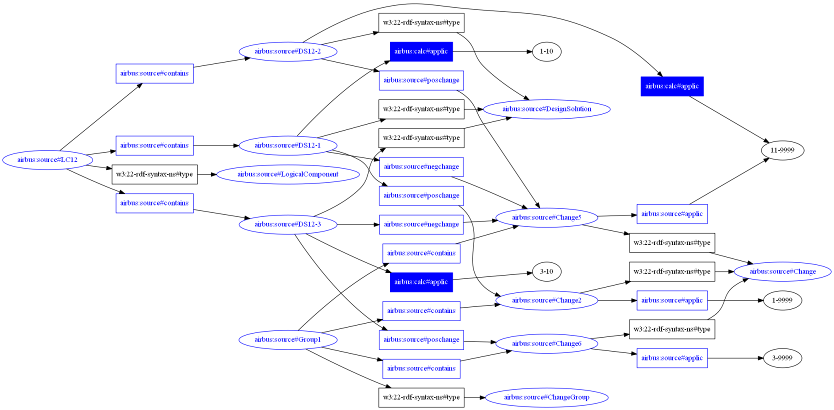
For each cell, we can generate (f_1):

- `(∧ (type` C_{ij} K_j `) (`K_j C_{ij} L_i `))`
- `Or (`K_j L_i C_{ij}`) when` C_{ij} `is a value`

Initial objects converted in RDF

The data are imported with a unique grammar, valid for many sources

Step 5: Making explicit implicit values on source data



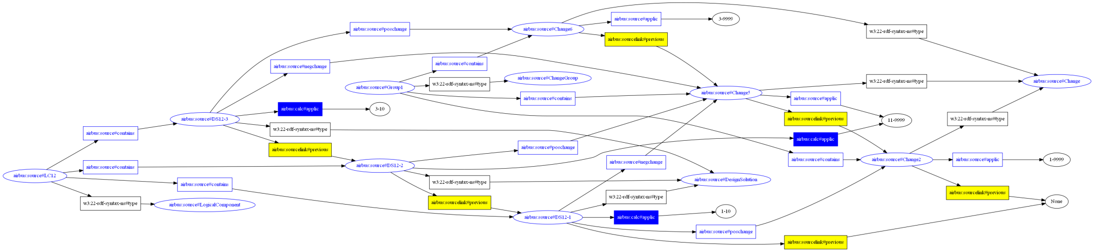

```
[Rule f2:
(type  ?ds DesignSolution)
(type  ?c1 Change)
(type  ?c2 Change)
(type  ?r1 SourceApplicabilityRule)
(applicability  ?ds ?r1)
(=  ?r1 (∧  ?c1 ¬?c2))
→ (applicability  ?ds
   (∩  (applicability  ?c1)
   (applicability  ?c2))) ]
```

f_2 applicable on DS12-1:

```
(type  DS12-1 DesignSolution)
(type  Change2 Change)
(type  Change5 Change)
(type  Rule1 SourceApplicabilityRule)
(applicability  DS12-1 Rule1)
(=  Rule1 (∧ Change2 ¬?Change5))
→ (applicability  DS12-1
   (∩  (applicability  Change2)
   (applicability  Change5))) ]
→ (applicability  DS12-1
   (∩  (∧  [1, ∞[ ¬[11, ∞[)))
→(applicability  DS12-1
   (∩  (∧  [1, ∞[ [1, 10])))
→(applicability  DS12-1 [1, 10])
```

Calculations are done based on implicit rules on existing data (here applicability)

Step 6: Making explicit implicit links in source data

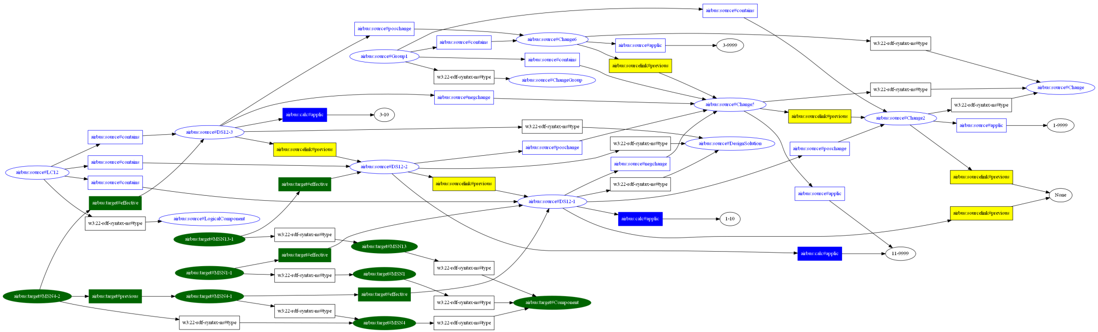


The original links between source data are completed when implicit

```
[Rule f3 first part:
(type  ?ds1 DesignSolution)
(type  ?ds2 DesignSolution)
(name  ?ds1 ?name1)
(name  ?ds2 ?name2)
(dash  ?ds1 ?dash1)
(dash  ?ds2 ?dash2)
(∧  (=  ?name1 ?name2) (=  ?dash2 (+  ?dash1 1)))
→ (sourcelink-previous  ?dash2 ?dash1) ]
```

```
[Rule f3 second part:
(type  ?c1 Change)
(type  ?c2 Change)
(change-number  ?c1 ?number1)
(change-number  ?c2 ?number2)
(=  ?number2 (+  ?number1 1)))
→ (sourcelink-previous  ?c2 ?c1) ]
```

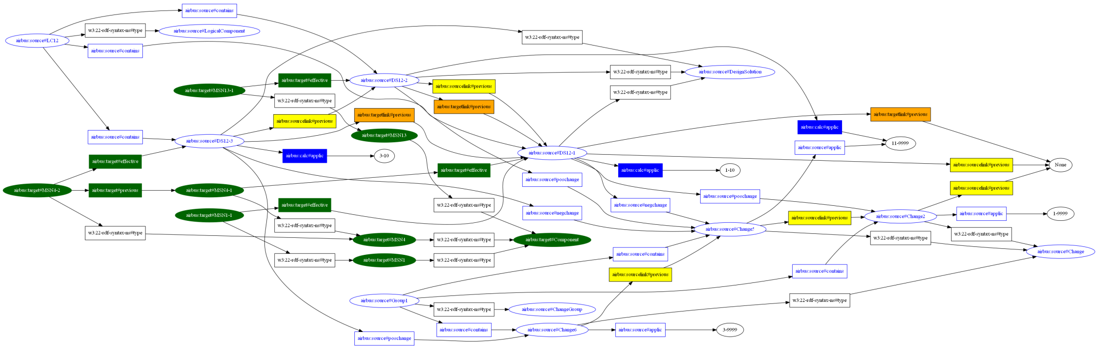
Step 7: Create new entities and link them to source data



New entities will be like new perspectives on data and will bring other member groups, so a new semantics

`Rule` f_4 `is creating new objects (MSNs) and is linking them to all DS effective to them.`

Step 8: Deducing new links between new entities



The new links complete the new semantic perspective (f_5)

Focus on the f_5 transformation

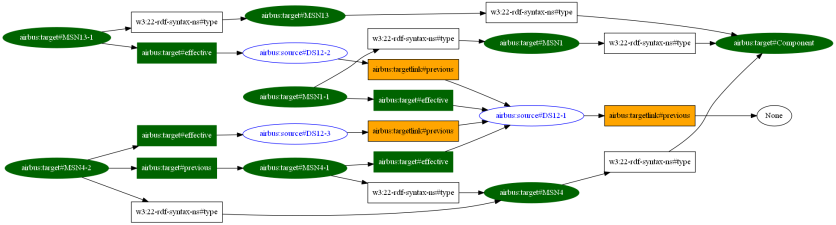
The f_5 rule uses several steps to infer a "fork" representation in the semantic space of new concepts

This transformation enables us

- To create a sensibly different chaining of design solutions, some being considered as forked out of the main history line (retrofit)
- To enable a new "interpretation" of the past data
- To start in the new system with a fully compatible history as if the past works would have worked in the new semantic space

```
[Rule f5:
(type  ?dx DesignSolution)
(type  ?da DesignSolution)
(creation  ?dx ?x)
(creation  ?da ?a)
(type  ?x Change)
(type  ?y Change)
(previous  ?x ?y)
(<  (applicabilityFrom  ?x) (applicabilityFrom  ?y) )
(type  ?n ChangeIndex)
(index  ?x ?n)
∃ ?p ?q:
(type  ?p ChangeIndex),
(type  ?p ChangeIndex),
(type  ?a Change),
(type  ?b Change),
(index  ?a ?p),
(index  ?b ?q),
(=  ?q (+  ?p 1)),
(<  (applicabilityFrom  ?a) (applicabilityFrom  ?x) ),
(<  (applicabilityFrom  ?x) (applicabilityFrom  ?b) )
→ (previous  ?dx ?da) ]
```

Filtering on the new perspective only



- The semantic perspective is completely new
- New entities point to unmodified existing entities (mainly “augmented”)
- The new model is adapted to the new system
- New ways of working are enabled by the shift in semantic perspective

Summary of steps and comments

#	Step	Graph transformation	Comment
f_1	Data migration to RDF repository	Multi-source data are split in RDF following a unified grammar	Semantic links are translated as they are Choice of semantic constructs is crucial
f_2	Doing calculation to make some value explicit	Data enrichment by calculation	-
f_3	Making implicit links explicit	Data enrichment by linking	Based on rules, missing links can be recreated
f_4	Create new entities	Data enrichment by creation	Semantic constructs must be chosen very carefully
f_5	Creating new links	Data enrichment by linking	Rules can deduce new semantic links from the previous steps

Important points

- Source data must be complemented to be as explicit as possible
- Triples are a good representation of fully split linked data
- Each transformation step is preserving the semantic topology of the initial graph (enrichment)
- Each step is a predictable function mixing business rule and topology pattern matching
- The data obtained by “reasoning” are possible to create once data are explicit
- The RDF repository is maintaining the audit trail between source and target data
- The formalization of transformation rules enables to ensure the quality of transformation

Conclusion

Conclusion

Current status on the project

Topic	Comments
Interactive semantic graph viewer	• It is crucial to be able to see and explore the semantic data • Ideally, we would like to interactively design graphically the graph transformation rules
Formalization of the "conversion ontology" (f_1)	• Currently implemented in a proprietary "command language" mixing conversion ontology and domain ontology, which is not satisfactory • Several grammar formats were tested (BNF, EBNF, RDF/RDFS-based) but we need to split both concerns and use the appropriate DSLs for each task
Formalization of conversion rules (from f_2 to f_5)	• We need to formalize more strictly the conversion rules, possibly with a powerful DSL • Description logics seem very interesting in formalizing the rules but we are just discovering the domain

Technical environment

- Python developments for CSV to RDF conversion (f_1)
- First wave of tests realized with Franz AllegroGraph (2018)
- Second wave of tests realized with Apache Jena (2019)
- Transformation rules currently coded in Python and mixed with Sparql queries (from f_2 to f_5)
- Currently used viewers: Graphviz, Franz Gruff, GML in Yed and Cytoscape
- Interactive semantic viewer ongoing works: Javascript, Node.js with adapted graph visualization libraries

Questions?

Symposium MaDICS 2019 – Reasoning on data

Journée `` Raisonnements sur les données : besoins applicatifs et techniques''

Jeudi 27 juin 2019, Rennes

Cet atelier commun avec l'action LEMON du GDR MaDICS a lieu dans le cadre du symposium MadICS

Programme

13h30-17h (horaires et ordre des exposés à préciser)

- **Olivier Rey** (Airbus) : Using semantic web technologies for aerospace industrial data migration
- **François Goasdoué** (IRISA) : Apprentissage de points communs entre données RDF et entre requêtes SPARQL
- **Nicolas Seydoux** (IRIT) : Impliquer les équipements en bordure de réseau pour raisonner à l'échelle du Semantic Web of Things
- **Valentina Dragos** (ONERA) : Pitfalls of social data exploration: Illustration on extremist content analysis

Détails des exposés

Olivier Rey (Airbus) : Using semantic web technologies for aerospace industrial data migration

The aerospace industry is managing aircraft products with PLM (product lifecycle management) software. Most of the time, the PLM software used are very customized and very old. Moreover in the day to day life, people are using a mix of various PLM systems and Excel spreadsheets. In order to change the business practices for the 3 core businesses of the aerospace companies (design office, manufacturing, support and services), the PLM backbones must evolve on two dimensions: first they must gather all data spread across various tools per business, and second they must integrate the three businesses together (digital continuity) in order to perform efficient concurrent engineering and gain on lead times. New end to end PLM backbones are available on the market but the migration to those products is slowed down by the lack of method and tools to properly migrate the data. The main constraint of PLM data migration is to be able to keep all the semantic links of past data into the new system, because the aircraft certification is tied to those links and data. At the intersection of all those constraints, the semantic web technologies (RDF/RDFS, Triplestores, SPARQL-based rules, etc.) can help Airbus to convert its core industrial data to migrate them into new generations of PLM systems.

The presentation will expose a concrete industrial case and explain the status of the works and the tool chain involved in those works.

Nicolas Seydoux (IRIT) : Impliquer les équipements en bordure de réseau pour raisonner à l'échelle du Semantic Web of Things

Le Semantic Web of Things est le domaine né de la convergence de l'IoT et du Web Sémantique. La motivation première pour le développement de ce domaine est le besoin d'interopérabilité qui naît de la grande hétérogénéité de l'IoT, autant dans les technologies mises en jeu que dans les domaines d'applications couverts. Cependant, intégrer les technologies et les principes du Web Sémantique dans l'IoT n'est pas trivial, du fait des importantes contraintes sur les ressources dans l'IoT, ainsi que du volume important de données à traiter, issues de systèmes fortement dynamiques et distribués. C'est pourquoi, en complément du Cloud computing, le calcul distribué sur les équipements en bordure de réseau, aussi appelé fog computing, a fait son apparition en tant qu'infrastructure pour le SWoT. Dans cette présentation, je présenterai mes travaux abordant la distribution dynamique de règles de déduction à travers un réseau d'objet connecté, s'appuyant à la fois sur les infrastructures Cloud et Fog.

François Goasdoué (IRISA) : Apprentissage de points communs entre données RDF et entre requêtes SPARQL
TBA

Valentina Dragos (ONERA) : Pitfalls of social data exploration: Illustration on extremist content analysis (titre provisoire)

TBA

About the author

Olivier Rey

- 25 years of experience in software companies and IT service companies involved in complex software projects
- +20 years in software R&D
- +10 years as a senior enterprise architect
- +10 years as a program director in complex projects
- In the aerospace industry since 2013
- Creator of the graph-oriented programming approach
- Expertise in high end distributed transactional system and middleware

olivier.o.rey@airbus.com | orey.github.io | github.com/orey