

10 years of Model Federation with Openflexo : Challenges and Lessons Learned

Jean-Christophe Bach, Antoine Beugnard, Joël Champeau,
Fabien Dagnat, **Sylvain Guérin**, **Salvador Martínez**

MODELS'2024, Linz, September 26th

Research project
Academic partners



Industrial partners



- ▶ A *model federation* framework
- ▶ A scientific experimental platform
- ▶ A successful but defunct company
- ▶ Many projects, academia and industry collaborations.





- ▶ Complex systems : multiple **points of view**
- ▶ Complex systems : heterogeneity (tools, actors, formalism)



- ▶ Complex systems : multiple **points of view**
- ▶ Complex systems : heterogeneity (tools, actors, formalism)
- ▶ We were (and remain) modeling enthusiasts



► C1: Preserving existing practices

- Preserve data and life cycles
- Do not break what already works (existing tools and processes)
- Do not require to replace tools or re-train users to new tooling



- ▶ C1: Preserving existing practices
- ▶ **C2: Seeing everything as a model**

- ▶ Any source of information may be considered as a model
- ▶ The "most appropriate" formalism
- ▶ Requires a way to interpret (or reinterpret) data contextually



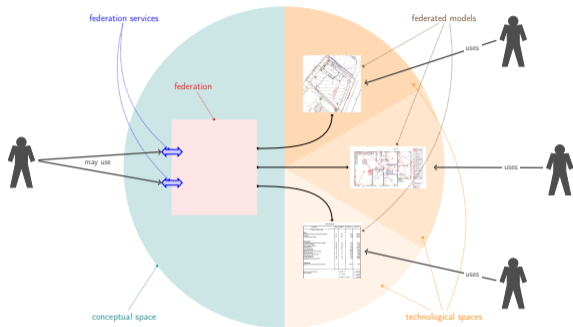
- ▶ C1: Preserving existing practices
- ▶ C2: Seeing everything as a model
- ▶ **C3: Conceptualizing and organizing**

- ▶ Find a way to reference elements source and define links
- ▶ Make possible for new concepts to emerge



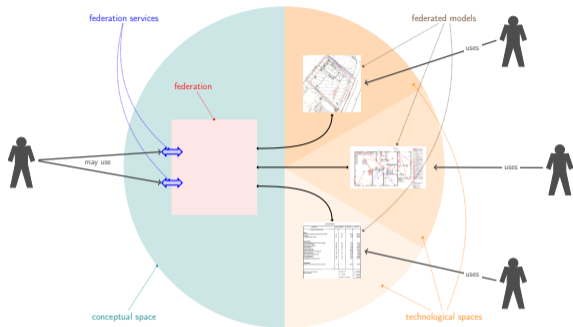
- ▶ C1: Preserving existing practices
- ▶ C2: Seeing everything as a model
- ▶ C3: Conceptualizing and organizing
- ▶ **C4: Mastering evolution**

- ▶ Detect (and eventually react to) changes
- ▶ Deal with/Tolerating inconsistencies



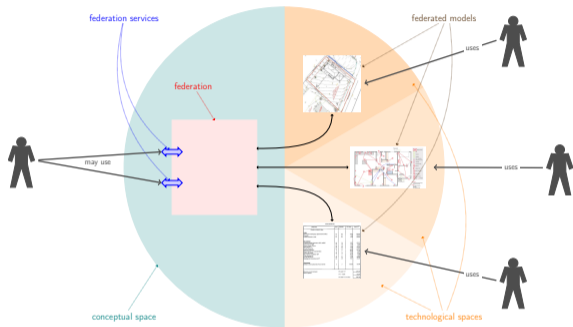
- ▶ Federated models
- ▶ Technological space

Federated models remain in their original **technological spaces**, users keep their practices



- ▶ Federated models
- ▶ Technological space
- ▶ Correspondence
- ▶ Conceptual space
- ▶ Federation

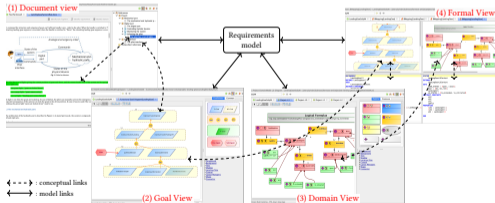
Their **correspondences** are reified within a model of the **conceptual space** called a **federation**



- ▶ Federated models
- ▶ Technological space
- ▶ Correspondence
- ▶ Conceptual space
- ▶ Federation

A **federation** is connected to a set of **federated models** and manages their **correspondences**. A **federation** can provide new services

► UC1 : Formose



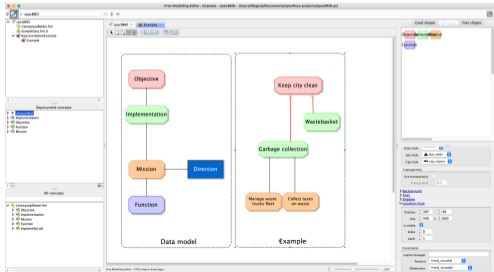
- A model-based requirement engineering tool
- Research prototype with academic and industrial partners

- ▶ UC1 : Formose
- ▶ **UC2 : SSE4Space**

The screenshot displays the SSE4Space project management interface. On the left, a sidebar lists various activities such as 'Address public perception issues and sensitive topics' and 'Provide security guidelines for operators'. The main area features a Kanban board with columns for 'Backlog', 'Ready', 'Done', 'Priority', and 'Not ready'. The 'Priority' column contains a task card titled 'Phase 2: Mission requirements identification'. Below the board, a detailed view of a task is shown, titled 'Characterize the security aspects of the solution space'. This view includes a dependency diagram with three nodes: 'Identify the security objectives and constraints for the project', 'Assess the security impact of the solution space', and 'Evaluate and select viable options'. A 'Status' dropdown menu is visible on the right side of the task view, with 'Not Ready' selected.

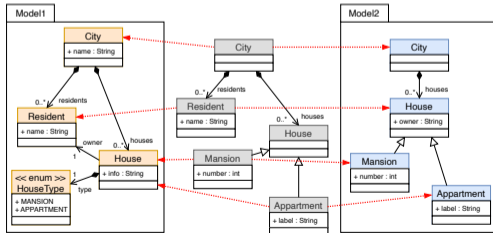
- ▶ Managing the security concern in satellite development
- ▶ An industrial project developed with Telindus for European Space Agency

- ▶ UC1 : Formose
- ▶ UC2 : SSE4Space
- ▶ **UC3 : Free Modeling Editor**



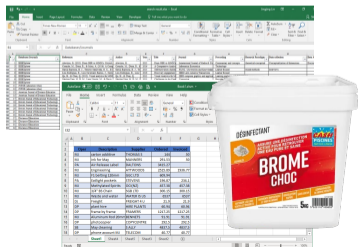
- ▶ Tool to co-construct a model and its metamodel
- ▶ Centered around an example diagram that may not conform to a metamodel

- ▶ UC1 : Formose
- ▶ UC2 : SSE4Space
- ▶ UC3 : Free Modeling Editor
- ▶ **UC4 : Model Mapping**



- ▶ Tool to build model mapping between various DSML
- ▶ Mapping, edition and consistency management

- ▶ UC1 : Formose
- ▶ UC2 : SSE4Space
- ▶ UC3 : Free Modeling Editor
- ▶ UC4 : Model Mapping
- ▶ **UC5 : PDF Labels**



- ▶ Tool to audit and update internal databases from information contained in PDF labels
- ▶ Federate informations using business-specific rules

- ▶ **L1: Separate business from technique**

- ▶ Focus to the essence of things and forget about technical details
- ▶ When modelling focus on business concepts, forget your own modeling habits
- ▶ A space for technical aspects, a space for domain-specific semantics

- ▶ L1: Separate business from technique
 - ▶ **L2: Tool maintenance and dissemination**
-
- ▶ Flexibility comes at a price
 - ▶ Keeping up to the technical stack is hard (e.g., Java)
 - ▶ Dissemination is not easy

- ▶ L1: Separate business from technique
 - ▶ L2: Tool maintenance and dissemination
 - ▶ **L3: Need for many concrete syntax**
-
- ▶ The most appropriate language for the right person at the right time :
 - ▶ Graphical form-based is ideal for small projects, textual is better for bigger and more complex projects
 - ▶ Diagramming is well suited at the early stages, textual is more appropriate for developers
 - ▶ ...

- ▶ L1: Separate business from technique
- ▶ L2: Tool maintenance and dissemination
- ▶ L3: Need for many concrete syntax
- ▶ **L4: Federation requires organization**

...as they grow rapidly !

- ▶ Designation problems are solved with technological adapters
- ▶ Federation models are models too and re-use model organization techniques

- ▶ L1: Separate business from technique
 - ▶ L2: Tool maintenance and dissemination
 - ▶ L3: Need for many concrete syntax
 - ▶ L4: Federation requires organization
 - ▶ **L5: Modeling requires flexibility**
-
- ▶ Free modeling help capture domains
 - ▶ Interpretation (vs code generation) improves productivity in early stages

- ▶ L1: Separate business from technique
 - ▶ L2: Tool maintenance and dissemination
 - ▶ L3: Need for many concrete syntax
 - ▶ L4: Federation requires organization
 - ▶ L5: Modeling requires flexibility
 - ▶ **L6: Modeling needs model capitalization**
-
- ▶ Re-use is key (especially for large scale modeling)
 - ▶ Federation eases re-use





Openflexo

- ▶ Collaboration
- ▶ Free Modeling
- ▶ Maintainability
- ▶ Digital Twins



Openflexo

- ▶ Collaboration
- ▶ Free Modeling
- ▶ Maintainability
- ▶ Digital Twins





Openflexo

- ▶ Collaboration
- ▶ Free Modeling
- ▶ Maintainability
- ▶ Digital Twins



- ▶ Want to..
- ▶ Collaborate?
- ▶ Contribute?
- ▶ Contact us!



Openflexo

- ▶ Collaboration
- ▶ Free Modeling
- ▶ Maintainability
- ▶ Digital Twins



- ▶ Want to..
- ▶ Collaborate?
- ▶ Contribute?
- ▶ Contact us!

Model Federation with Openflexo

Why model federation?

- Modeling an overall system requires to connect sequentially or several systems of one or a group of sites composed to specify processes in terms of processes, events, tasks...
- These sites are heterogeneous, use different of the system, use the set of several cases...
- We have to manage that heterogeneity / heterogeneity without changing existing practices

What is model federation?

- Federated models create in their original technological spaces, users keep their practices
- They interoperate on virtual federated models of the common interest of all federates
- A federator is connected to a set of federated models and manages their interdependencies
- A federator can provide new services

Model federation with Openflexo

- Openflexo is an open source software framework supporting model federation
- It relies on a 2008, the model federation (FM) (Federation Modeling Language)
- Openflexo libraries are implemented/Developed/Implemented and/or managed by FM.
- It includes libraries to access to federated models, added on existing systems...
- The main technology options provided are FM, XML, Java, Java Script, PHP, etc., and FM.

FM, structure a federator using virtual models that create concepts

- It is a 3rd level level accessible about external modeling language
- Concepts (i.e. shared feature primitive (i.e. attributed and instance (i.e. methods))
- Primitive data rules to federated models using model rules defined by technology solutions
- Behavior defines to an expressive language how external resources are managed
- FM supports distributed, reconfigurable, multiple federations, adaptability, reformation...

3 application examples

- Farned**
 - 4 model-based requirements engineering tool
 - Federation connecting 10 FM, 27 systems (1 000 lines)
- SSEEspace**
 - Managing the energy systems in specific infrastructure
 - Federated security infrastructure with process federation
 - Federated with "Model for Business Space Energy"
- Free modeling**
 - Tool to generate and simulate models and scenarios
 - Control around an energy system that may not be subject to constraints
 - Provides actions on the model, its metadata and their representation...