Why incomplete diagrams are useful in reaching a common vision

- **Date**: 29th March 2013
- **Author:** Guillaume FINANCE, Objet Direct Analyst & Consultant

You may wonder why an incomplete or incorrect diagram would be useful. Shouldn't it make sense to aim at producing a good representation right from the start by modelling what needs to be built and achieved?

This article deals with a simple approach: **incomplete diagrams provide a visual representation of the system to be built, that once shown to stakeholders, product owners, dev teams, and project managers, will trigger feedback and discussions to move towards a shared vision.** This approach is compatible with iterative and agile projects where a diagram is not intended to be static and unchanged, but to be updated throughout iterations. In some cases, a diagram will have a single use to let the team agreeing on what needs to be achieved with no intent to carry updates in this diagram for any further use (concept of a "throwable diagram").

The first version of a model or diagram is probably incomplete as requirements often change throughout a project (proof of concept, prototype, business entity model, etc.). A good exercise to receive feedback is by organizing a workshop, where the initial version of the diagram should prompt reactions from each team member, whether they have a business or technical opinion. Updates can be carried in a "live mode" on the UML, SysML or BPMN diagrams via the chosen modelling tool such as **Sparx Enterprise Architect**, or later on when all discussions are over. In either situation, the workshop will lead to a single vision and statement of the problem, acknowledged by the team.

Models and diagrams enable us to visualize the problem, state and discuss our views or opinions, and hence iteratively build a vision that's agreed, common, and shared by all. This will require more or less iterations and efforts depending on the context and specific needs of the project.

Updating models on a regular basis provides the benefit of delivering a set of diagrams and specifications on which the team can rely on, especially after the product delivery to assess impacts related with bug fixes, change requests, or to carry the product's maintenance.

Experience and benefits:

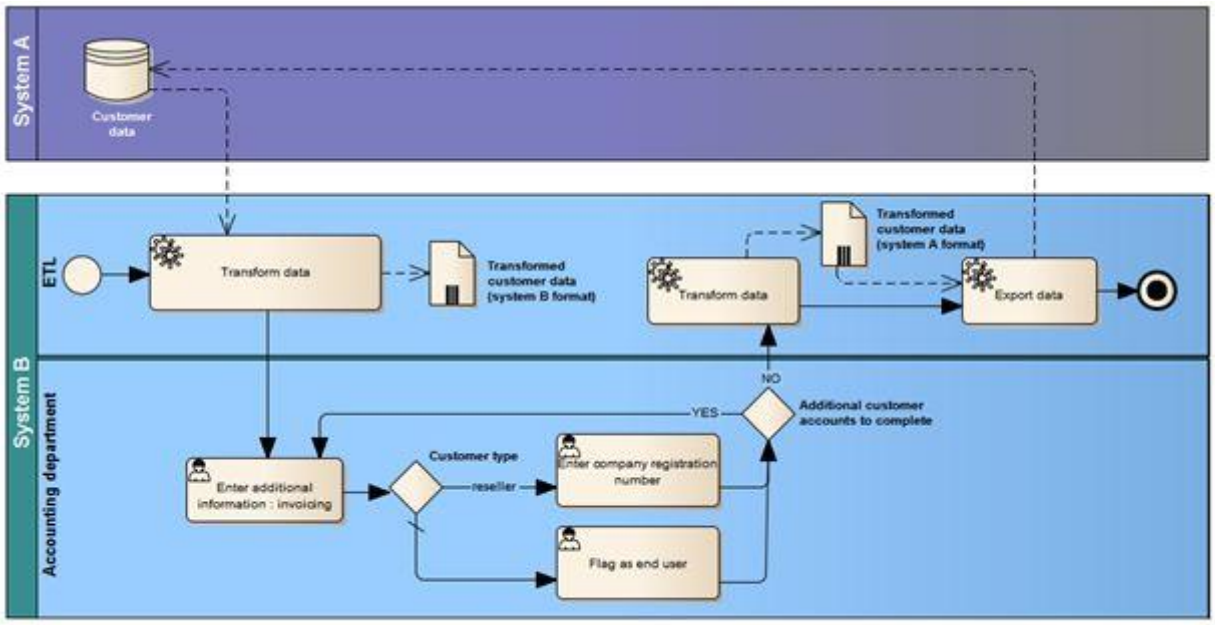
I recently got an email from a colleague to whom I sent **an incorrect BPMN diagram from a business perspective**, but correct from the BPMN semantics perspective. This diagram was aimed at capturing a full process to discuss a GO/NO-GO decision; he told me that "the incorrect diagram has already made it possible to identify issues and make progress on the shared vision with the stakeholder". This initial feedback has easily and promptly been captured in a revised version of the BPMN diagram.

I also once had the task to define a functional model by identifying information flows between two interacting products, implemented and maintained by separate teams. Each development team provided me with extensive information supporting their respective architectural and technical choices. Each team had more or less an idea of how the interacting product was working. UML component and deployment diagrams refined through iterations not only let each team have a common and final view over the mechanisms for their product, but also access to the corresponding accurate view of the interacting product.

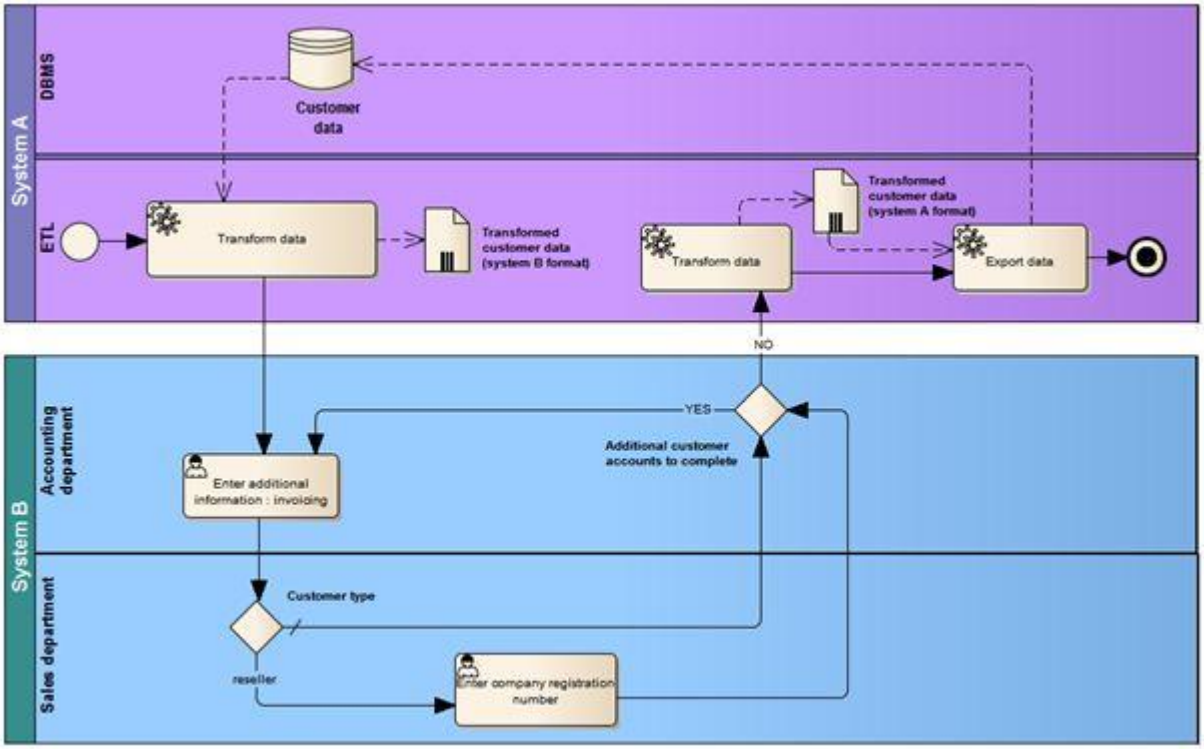
Through the use of an "obviously" incomplete diagram, numerous exchanges and conversations can be triggered, and lead to a global and shared representation of a problem or an implemented product, e.g. to support business, technical or architectural decisions to evolve it.

Below is an illustration of the identified changes required from the initial version of a BPMN diagram:

- ETL activities are carried by "System A" instead of "System B"
- A Sales Department and its responsabilities have been identified



Iteration 0


System A
DBMS
Customer data
ETL
Transform data
Transformed customer data (system B format)
Transform data
Transformed customer data (system A format)
Export data
NO
System B
Accounting department
YES
Additional customer accounts to complete
Enter additional information : invoicing
Sales department
Customer type
reseller
Enter company registration number

Iteration 1