A Quick and Dirty Yaca Diagramming Tutorial

1 Introduction

Yaca-based process diagrams (a.k.a. **Yaca diagrams**) are means to describe all kind of business processes in organizations but more specifically those where collaboration tasks are crucial in their success. By collaboration tasks, one shall understand these tasks involving together several members of an organization or of partnering organizations with two possible objectives: either sharing information so that, at the end of the collaboration task, participants own more knowledge, or sharing information with the aim of making a collective and consensual decisions. Tasks of this nature are common in organizations, more specifically in activities of entrepreneurial nature (research and development activities, governance and managerial activities).

The Yaca process-modelling notation (**Yaca modelling** in short) is strongly inspired from its predecessors and does not pretend to replace them. It just exists to complement some of these modelling notations (BPMN, YAWL, EPC to cite just a few[1]) that are mainly used to specify information systems that are intended to support operations activities: production and servuction activities, administrative activities for instance. In other words, BPMN will still be the appropriate tool to model the interactions of users with information systems, such as an ERP or a CRM, and to capture the user requirements towards these systems. Yaca modelling is likely to be more efficient to capture meta-processes "as a whole" and not limited or focussed to the interactions of "operational users" with the information systems.

Yaca diagrams are primarily intended to model engineering and engineering management processes which are the core of technical projects and programs. By extension, Yaca diagrams are suitable means to describe study, project and program processes as a whole, including technical processes such as requirements engineering, project costing, project planning and scheduling, project risk management, project controlling and reporting, configuration management, etc.

> **Yaca**Lite is an open-source SaaS ('Software as a Service') that can be used to prepare Yaca diagrams. A still *quick and dirty* but comprehensive tutorial to this software is given in section 5. However, some tips to use it efficiently are given in sections 2 and 3.

2 Yaca Diagram Vocabulary

A Yaca diagram is made of inter-connected components. Components are basically of six types:

- **Shapes** that depict actions: practically the **tasks** (**steps**) of the process;
- **Nodes** that state **events** such as the start and the end points of the process, but also so-called pause and rewind events;
- **Lanes** that are associated to **roles**, i.e. the **contributors** to the process;
- **Links** that set the **sequence** (**flow**) of tasks;
- **Media** that provide information on the **data** that is required or that is created by a task;
- **Annotations** that provide additional information to ease the interpretation of the Yaca diagram or means to refer to some of its components.

Note that for sake of simplicity, **decisions** are assimilated to tasks while **branching** (split and merge) is a concept associated to task sequencing.

2.1 Tasks (Shapes)

Tasks can be grouped into three families (see **fig. 1**).

Figure 1. Families of tasks.

Ordinary tasks. These tasks aim at transforming straightforwardly inputs into outputs. Inputs and outputs can be of

[1] See `www.bonnal.eu` for additional resources on these process modelling frameworks.

tangible nature (raw material, equipment, system, facility, etc.) or intangible (knowledge, data, etc.). Documents and drawings are certainly in between. Ordinary tasks are drawn as rectangles. They are available with different icons to ease the reading of Yaca diagrams (see **fig. 2**).

Figure 2. Ordinary tasks.

Decisions. These tasks performed by contributors aim at transforming information into decisions. They are drawn as rhombs (see **fig. 3**).

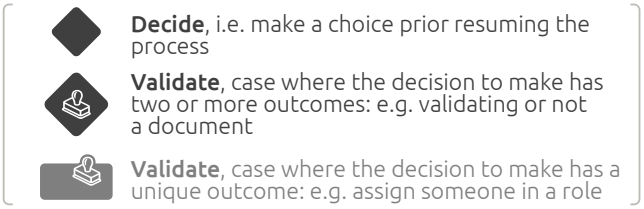

Figure 3. Decisions.

Collaboration tasks. These tasks necessarily involve several contributors synchronously, mainly to transform individual knowledge into shared or collective knowledge. They are drawn as circles but also as rhombs if they are associated to decisions (see **fig. 4**).

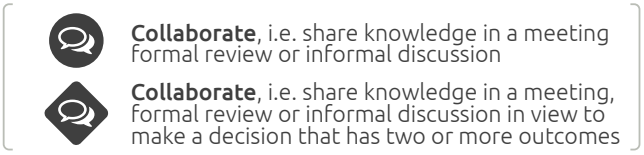

Figure 4. Collaboration tasks.

All tasks shall be labelled; the label can be located anywhere around the shape. Labels shall typically made of an action verb (infinitive tense) followed by a substantive (see **fig. 5**). Provisions such as the ones given in MIL-HDBK-245D Appendix B.2[2] or ASD-STE 100[3] can advantageously be followed to select non ambiguous action verbs.

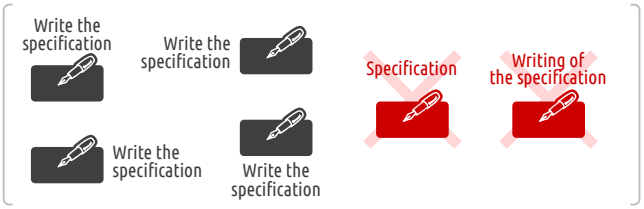

Figure 5. Task labelling.

> **TIP • YacaLite:** Use the palette button highlighted on **fig. 6** to reposition the task label around its shape.

Figure 6. YacaLite palette – Label positioning.

2.2 Events (Nodes)

Any process shall have a starting point (source) and an end point (sink); they are referred to as the **start event** and the

[2] US Department of Defense, *Preparation of Statement of Work*, MIL-HDBK-245 rev.D, Washington, DC, USA, 1996.
[3] Aerospace and Defence Industries Association of Europe, *Simplified Technical English*, ASD-STE 100, www.asd-ste100.org.

end event (see **fig. 7 (a)** and **(b)**). These shapes were designed based on YAWL equivalent components.

To augment the expressivity of Yaca diagrams, two additional event types were appended: a so-called **pause event** and a so-called **rewind event** (see **fig. 7 (c)** and **(d)**).

The **pause event** can be used to terminate a sub-branch to avoid featuring a long overlapping arrow from the pending sub-branch to the end event of the diagram (which shall be unique). The modeller shall ensure that this pause event necessarily occurs prior the overall process is completed.

The **rewind event** can be used to interrupt and "rewind" part of the process when something has gone wrong. In sake of simplicity, it is let to the contributors of the process to define where back in the process to restart.

Examples of use of these two specific events are given in section 3.

Rule: A given Yaca diagram can feature one start event and one end event only. However, several pause and rewind events can be featured.

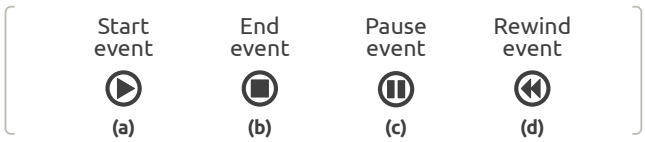

Figure 7. Typology of events.

2.3 Contributors (Lanes)

The contributors (i.e. the participants) to a process are identified by means of lanes (see **fig. 8**). Except collaboration tasks that can be drawn outside lanes, all tasks performed by a contributor shall be drawn in the corresponding lane. Lane labels, i.e. contributor role, are featured on the right hand side, vertically.

It is likely that several contributors take part to a process; their lanes are just drawn one below the other. There is no specific rule to set the order of lanes, but the modeller shall order them so that the network of arrows featuring the sequence is *optimized*.

The concept of lane is borrowed from BPMN.

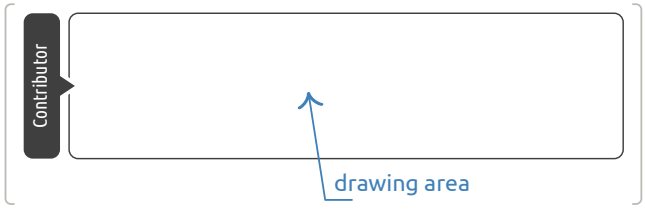

Figure 8. Lanes.

2.4 Routing and Branching (Links)

By definition, a process consists of a sequence of steps, i.e. of tasks in Yaca diagramming. This sequence is set by means of links, i.e. of oriented arcs (see **fig. 9**).

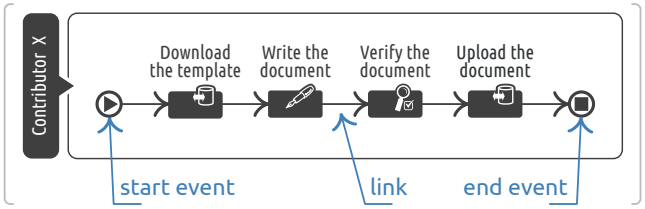

Figure 9. Sequence of tasks.

Links emanating from regular tasks are *regular links* (see **fig. 10 (a)**), their heads feature an arrow and they have nothing specific at their tails.

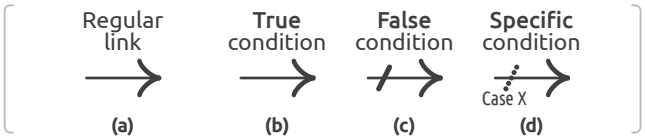

Figure 10. Typology of links.

Links emanating from decisions (i.e **conditional routing** or **disjunctive branching**) can be of three types. If the condition is true, then the link is a *regular link* (see **fig. 10 (b)**). If the condition is false, in addition the link features an oblique solid bar at its tail (see **fig. 10 (c)**).

There are also decisions where several cases might be considered. The links emanating from such decision rhombs have oblique dashed bars at their tails, complemented with a short description of the condition associated to them (see **fig. 10 (d)**). **Fig. 11** shows two equivalent Yaca implementations of links emanating from a decision task and **fig. 12** shows a Yaca implementation of a decision task leading to several cases.

Figure 11. Equivalent disjunctive branchings.

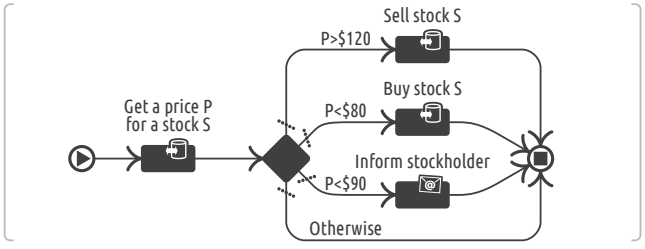

Figure 12. Example of a multicase decision implementation.

So-called **synchronization links** complement the typology of links (see **fig. 13 (a)**). They can be used when two (or more) tasks shall be performed by two (or more) contributors in a synchronized way, i.e. when one task needs information

from the other(s) to progress and vice versa. **Fig. 13 (b)** and **(c)** give two examples of use of synchronization links.

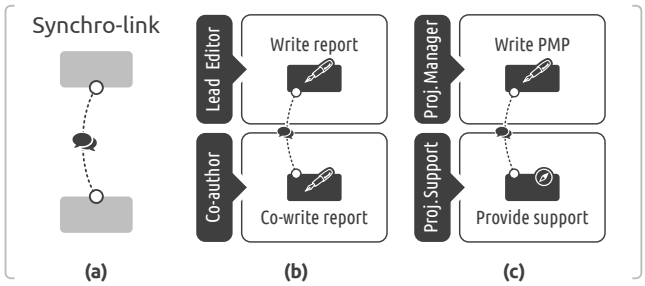

Figure 13. Synchronization link.

Parallel routing (or **conjunctive branching**) is also an important feature of any process diagramming language. Yaca diagramming provides two representations of parallel routings: the formal representation (see **fig. 14 (a)**) and a simplified or informal representation (see **fig. 14 (b)**).

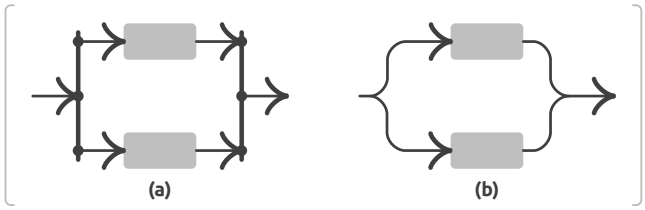

Figure 14. Equivalent conjunctive branching.

So-called **collaboration bars** are required to build collaboration tasks. Collaboration constructs are made of a collaborative task (see **fig. 4**) positioned outside the lanes—in between two lanes for instance—and of a vertical *bold line* crossing lanes and featuring a *bold dot* in each of the lanes participating to the collaboration task (see **fig. 15**).

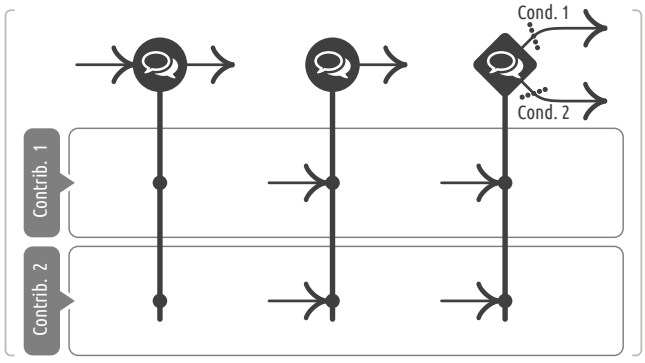

Figure 15. Construct of collaboration tasks.

2.5 Media

To ease the interpretation of a Yaca diagram, it might be useful to specify which are the **media** that are impacted by a task, either because it is required as an input or because it is created as an output. To address the needs of technical processes, Yaca modelling proposes four types of media (see **fig. 16**). **Fig. 17** shows a Yaca implementation featuring media.

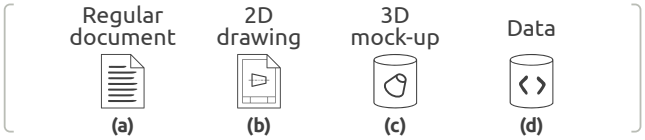

Figure 16. Typology of media.

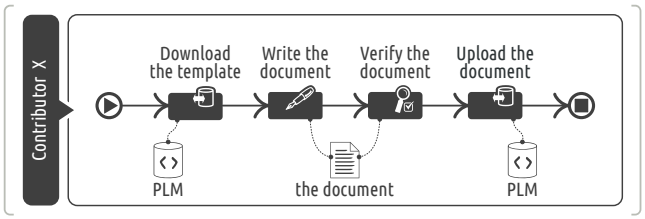

Figure 17. Yaca diagram featuring media.

2.6 Annotations

Annotations are additional information that is added to main components of Yaca diagrams to ease their interpretation. It is understood that for the sake of readability, annotations shall be used sparingly (see **fig. 18**).

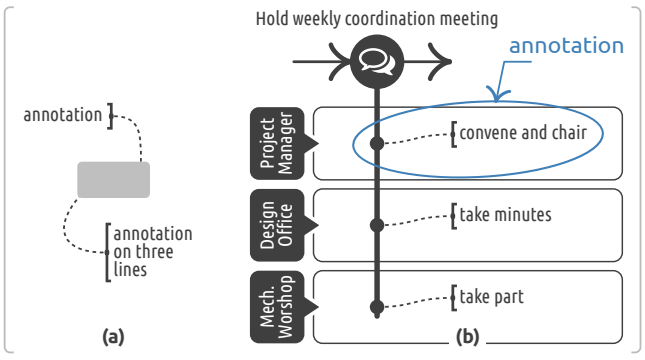

Figure 18. Annotations.

3 Yaca Diagram Syntax

Building a Yaca diagram does not consist of just locating components next to the other. Even if Yaca modelling were designed to leave some freedom to creativity, some rules shall be followed. The robustness of a process-modelling framework would be perfect if, for a given modelling problem, two modellers would build the same diagram. Of course, Yaca modelling is not strict enough to ensure such a robustness level; nevertheless, some syntax rules shall be followed to ensure that a Yaca diagram is drawn consistently and can be interpreted almost without ambiguity.

Simulating the progress of a **token** throughout a diagram is an approach that is rather effective to guarantee the consistency of a process diagram. In the context of Yaca diagramming, the rules for the **creation**, **transfer** and **absorption** of tokens are the following:

Rule 1. The process starts with the creation in a start event node of one token (or as many tokens as there are links departing from the start node; one per link) (see **fig. 19**).

Figure 19. Token creation in a start event.

Rule 2. Each task absorbs tokens at the rate their arrive; after a certain delay has elapsed corresponding to the task duration, each task creates one token (or as many tokens as there are conjunctive links or eligible disjunctive links departing from it; one per link)(see **fig. 20**). See also rewind event rule below.

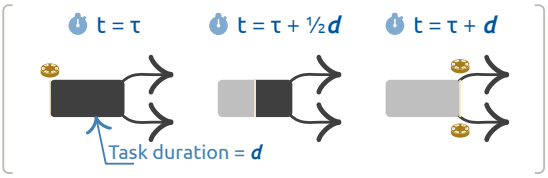

Figure 20. Token absorption-creation in a task.

Rule 3. Each link transfers instantaneously one token from the incoming component (task or event) located at its tail to the outgoing one located at its head (see **fig. 21**).

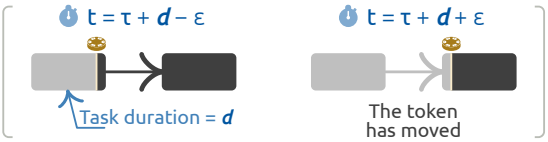

Figure 21. Token transfer in a link.

Rule 4. Each conjunction bar puts in a queue incoming tokens and when there are at least one token per incoming link, absorbs these tokens and creates instantaneously as many token as there are departing links; one per link (see **fig. 22**).

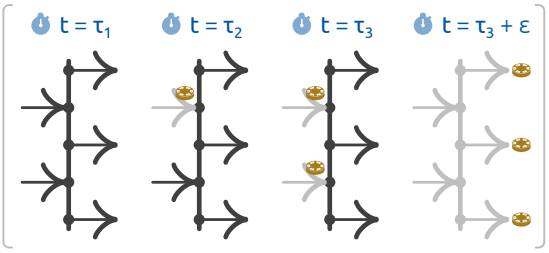

Figure 22. Token absorption-creation in a conjunction.

Rule 5. Each end event and pause event absorb tokens at the rate they arrive (see **fig. 23**).

Figure 23. Token absorption in an end event.

Rule 6. Each rewind event absorbs tokens at the rate they arrive and allows process contributors to decide where to create a tokens.

Rule 7. Media and annotations have no effect on tokens.

Rule 8. The process is said completed when there are no more tokens progressing, i.e. in process, into the flow diagram.

Rule 9. A process for which all the tokens cannot be removed is said not viable and shall not be validated! So the importance of quickly simulating token propagation to ensure the feasibility of the described process.

4 Examples of Yaca Diagrams

With version **0.2** of this document.

5 YacaLite Tutorial

With version **0.2** of this document.