Readable Class Diagram

Author: Mariusz Trzaska, https://www.mtrzaska.com/

This guide is intended to help in preparing a readable class diagram. Most of the remarks presented here can also be applied to other types of diagrams (activity, state, use case).

Substantive Correctness

First, ensure that the diagram is substantively correct—that is, that its contents (classes, attributes, associations, inheritances, etc.) are properly composed. How to achieve this is described in great detail in my book **[1]**. The class diagram that we will work with has also been taken from that book. The book and diagram are in Polish, but this does not significantly affect the tutorial.

Notation

Most diagrams (including class diagrams) are created according to a specific notation. By this we mean a set of symbols, their appearance, and their semantics (meaning). In the case of class diagrams, the most commonly used notation is UML.

When choosing an editor for drawing diagrams, make sure that it contains **the correct and required set of symbols**. This may seem obvious, but quite often one can encounter certain deviations that make understanding the content much more difficult.

Diagram Layout

Figure 1 shows an example class diagram taken from my book ([1]). We will not discuss why it contains these particular classes, attributes, associations, or inheritances - this is explained in detail in the mentioned book. Despite containing around 20 classes, it is quite readable and it is easy to understand what it includes. This is due, among other things, to the consistent use of certain conventions:

- Inheritance is drawn vertically, with the superclass at the top and the subclass at the bottom.
- Associations are placed horizontally so that they are always located on the left or right side of a class. For example, the association between “Publisher” and “Edition”, although partly running vertically, begins and ends at the side of a class.
- Associations include names and multiplicities.
- Lines are drawn vertically or horizontally, avoiding intersections.

We also try not to divide diagrams into multiple parts. This should only be done in very complex projects when the number of elements genuinely cannot fit on a single page.

Of course, applying the above principles is not always simple and may sometimes require, for example, changing the position of certain classes.

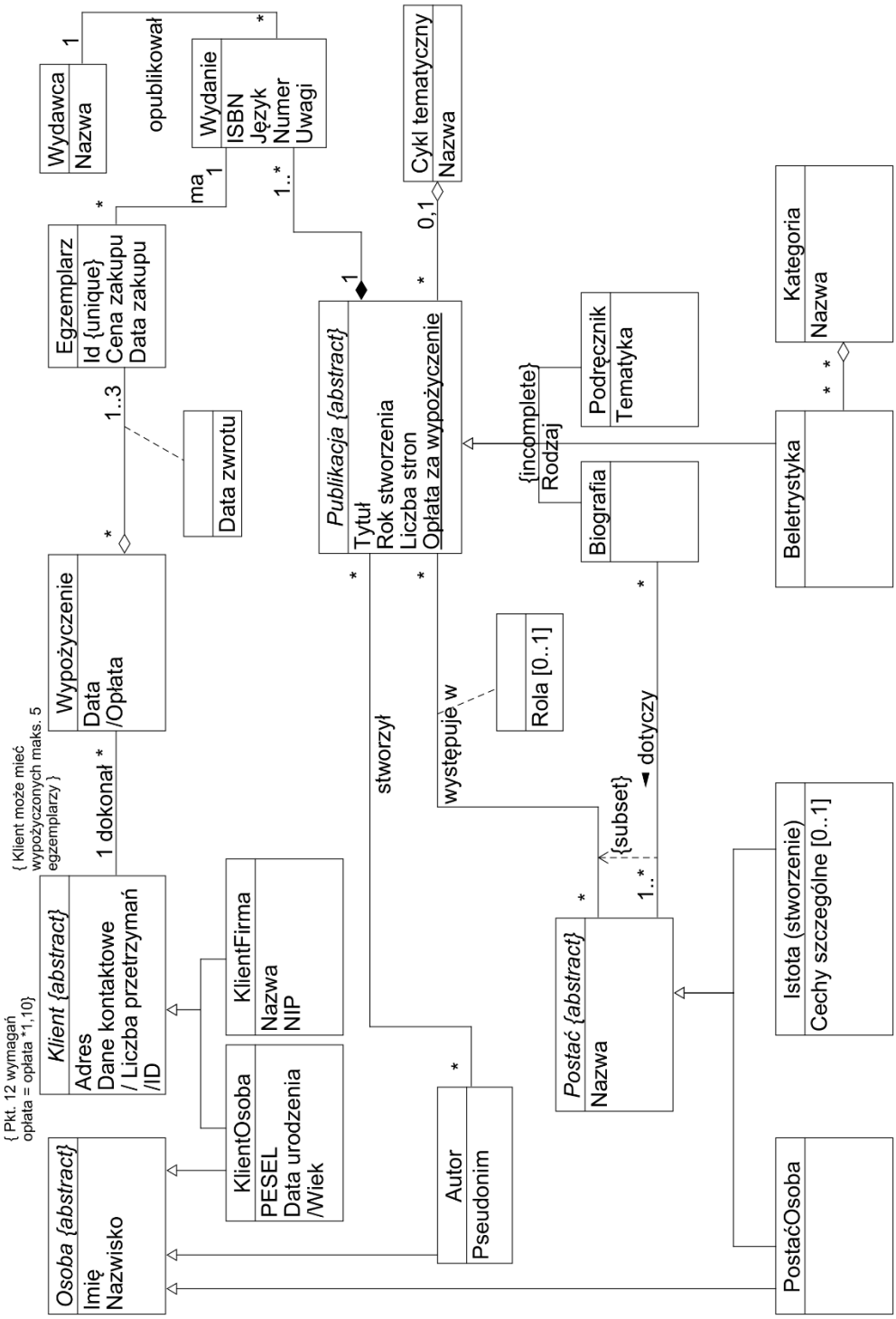

Figure 1. Sample class diagram. Source [1]

Diagram Proportions

To make optimal use of the available workspace, you should first determine where the diagram will ultimately be placed. Often this is an A4 sheet with dimensions 210 × 297 mm or 297 × 210 mm

(depending on the orientation: portrait or landscape). This brings us to another important issue. The actual page size is not as important as the **proportions of the page**. For A4, the ratio is approximately 1:1.41. Therefore, we should try to ensure that our diagram has similar proportions. What determines the proportions of a diagram? Clearly, its outline, which results from the placement of its outermost elements.

Let us look at Figure 2. It contains simplified visualizations of an A4 sheet with three variants of a diagram. All of them present the same class diagram, created in the same tool (UMLet [2]) and using the same font size. Why, then, is version (a) the most readable, while the others are not? What distinguishes them? The only difference is the proportions of the diagram, resulting from the **arrangement of its elements**. In (b), the right part of the diagram is placed too far from the rest, while in (c) the lower part clearly sticks out. As a result, in order to fit both the height and width, we must leave empty spaces on the sides. We can also look at the entire A4 page and notice that in (a) the shape of the diagram most closely resembles the shape of the page itself.

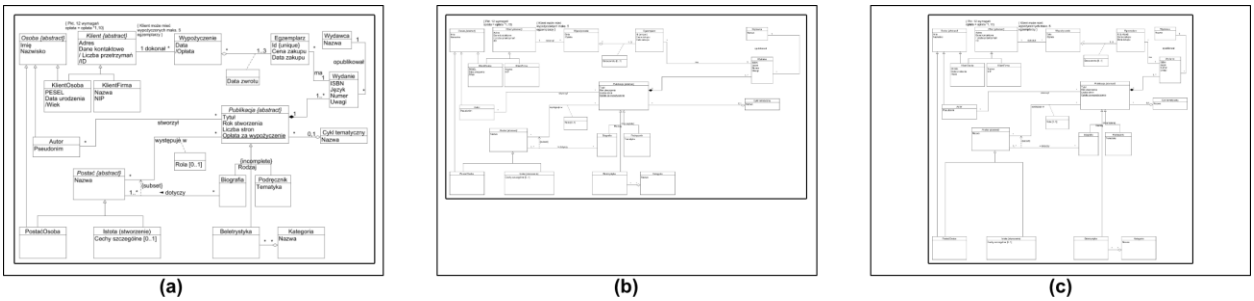

Figure 2. Diagram proportions and their readability on A4 pages

We should also be careful not to go too far in the opposite direction. We do not want to produce a maximally “compressed” diagram. There must be sufficient space between classes to draw connections and to place clear labels, such as multiplicities in the appropriate positions.

Naturally, attempting to graphically stretch an already inserted image within a document will distort the appearance of the entire diagram, the shape of the text and symbols, and ultimately **degrade its quality**.

Font Size

Another element that must be chosen properly is the font size used to describe elements in the diagram, such as class names, attributes, and so on. The key point is not simply to always use a large font, but rather to ensure that the font size is appropriate in relation to the other elements.

Figure 3 shows the diagram we already know in two variants. Both are identical (including layout and proportions), but they differ in the font size used. In the UMLet program, variant (a) uses the value 26, while (b) uses 12. Of course, it is not worth focusing on the exact numer - different programs interpret these values differently.

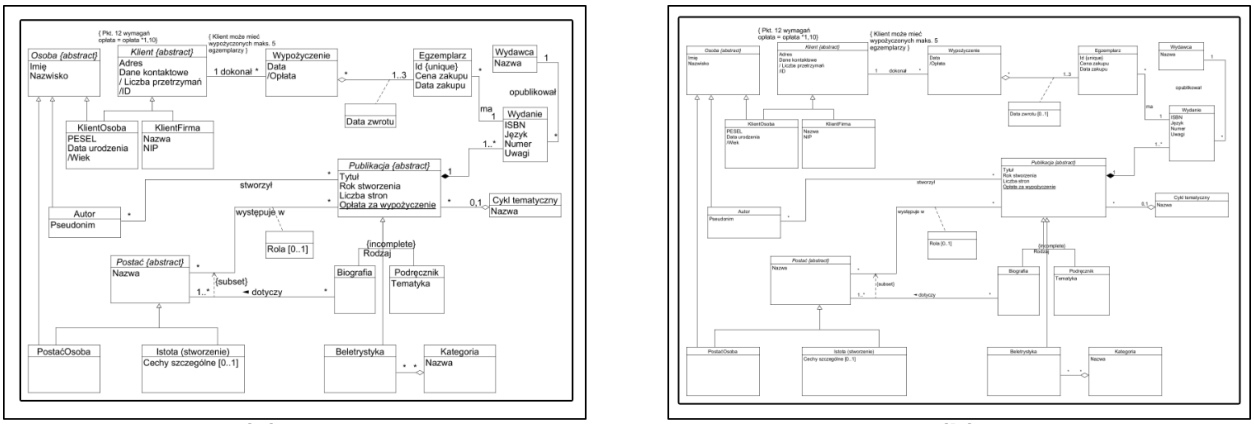

Figure 3. Font size and the readability of a diagram on an A4 page

We should also make sure that the **symbols** used in the diagram (e.g., the triangle used in inheritance) are **clearly visible** relative to the font size, class borders, and other elements. In some programs these symbols may be disproportionately small, which can make it difficult - or even impossible - to distinguish between, for example, composition, aggregation, and inheritance.

Exporting the Image

A diagram created in a dedicated editor must somehow be placed in the final document. To do this, we can export it to a common graphic format. Although the process itself is simple, several important aspects should be considered:

- File format. In general terms, images can be stored in either vector or raster (bitmap) formats. Vector formats (e.g., EPS, SVG) are generally better, but they are not always available (either for export or supported by the document editor). Alternatively, a bitmap format can be used - preferably with sufficiently high resolution and lossless compression (e.g., PNG). JPEG files should be avoided, as they are intended primarily for photographs.
- Avoid using screenshots as the source of the diagram image. Screenshots limit the image resolution and negatively affect the overall quality. This happens because the computer modifies graphics in various ways when displaying them on the screen.
- Resolution (expressed in pixels [3]) of the generated raster image. In general, the higher the resolution, the better. However, beyond a certain point (related to the pixel density of the target device - DPI [4]), increasing it further is unnecessary. As a rule of thumb, one side of the image should be between 2000 and 3000 pixels when it is intended to be placed on an A4 page. Figure 4 shows a fragment of a diagram exported as a high-resolution image (a) and a low-resolution image (b).
- Color scheme. The most readable diagrams use dark (black) lines on a light (white) background, as in the examples shown. Avoid using many colors or a dark (black) background.

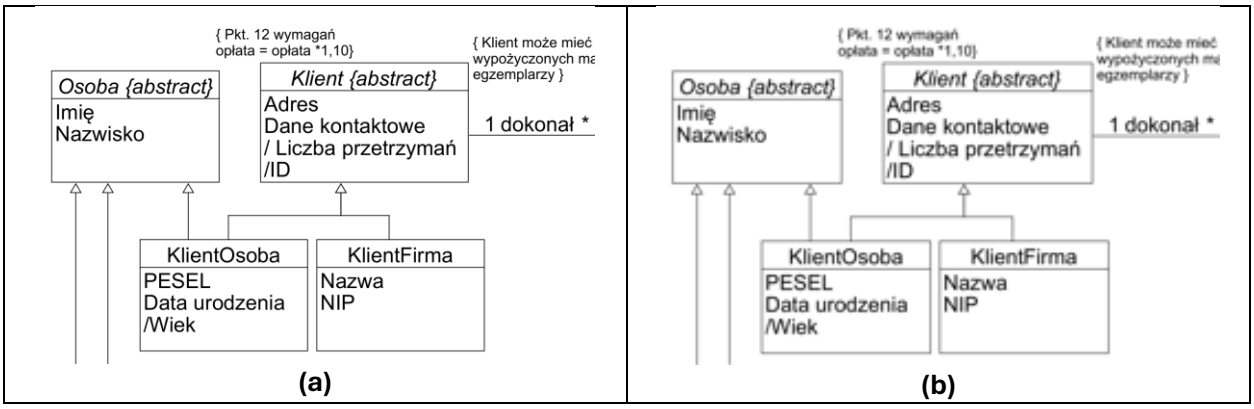

Figure 4. Image resolution and diagram readability

Summary

To prepare readable diagrams, we must pay attention to several elements that significantly influence what ultimately appears in the final document:

- Substantive correctness,
- Proper use of notation,
- Arrangement of diagram elements (classes),
- Image proportions matched to the proportions of the target document,
- Font and icon sizes used in the diagram,
- The method of exporting the diagram from the graphical tool to the target (text) editor.

I hope that this document will help in preparing readable diagrams that can be easily understood and conveniently modified whenever necessary.

References

[1] Mariusz Trzaska: Modelowanie i implementacja systemów informatycznych 2.0. Year 2025. Stron 487. ISBN 978-83-976442-0-5. *https://www.mtrzaska.com/ksiazka-mas-20/* Access: 2026-02-27
[2] UMLet Edytor diagramów. https://www.umlet.com/ Access: 2026-02-27
[3] Pixel (Wikipedia) https://en.wikipedia.org/wiki/Pixel Access: 2026-02-27
[4] Dots per inch (Wikipedia) https://en.wikipedia.org/wiki/Dots_per_inch Access: 2026-02-27