Smartdiagram: The Package and Its Journey

Claudio Fiandrino

Abstract

Smartdiagram born as a response to a question on TEX.stackexchange. The challenge was to emulate a feature that Microsoft Power Points provides: the capability of automate with animations a diagram. This feature allows to create diagrams from lists, so the user interface had to be as simple as possible, i.e., a list. In this article, I review the basic idea that overcomes the challenge and I expose the main features of the package along with a bit of its history.

Sommario

Smartdiagram è nato come risposta ad una domanda apparsa su TEX.stackexchange. La sfida proposta era emulare il comportamento di una funzionalità di Microsoft Power Points, ossia la capacità di creare in modo automatico un diagramma grafico da una lista di elementi con capacità di animazione dei singoli elementi. In questo articolo, si spiega l'idea di fondo in grado di vincere la sfida e si espongono le principali caratteristiche del pacchetto con una nota sull'evoluzione storica.

1 Introduction

Smartdiagram has born as a response to a question on TEX.stackexchange[1]. The challenge was to emulate a feature provided by Microsoft Power Points: the capability of creating a diagram from a list with animations. As LATEX does not share the What You See Is What You Get (WYSIWYG) paradigm, it is not possible to first write the list and then click to obtain the diagram. Nevertheless, to emulate the intuitive usage, the smartdiagram user interface transforming a list into the diagram had to be as simple as possible, i.e., characterizing intuitively the list, the type of diagram and the options to customize the diagram.

Based on Ti*k*Z Tantau (2010), the code employed in such answer has become the core of the smartdiagram package Fiandrino (2012). Along the subsequent months, many additional components have been included such as the support for the key-management interface, new types of diagrams and correction of (many) bugs and typos.

The package overcomes the initial challenge. However, despite its capability of full customization of the diagram provided by the key-management interface, smartdiagram limits somehow the user. Indeed, if one wants to move away from the predefined type of diagrams or do substantial modifications that break the automatism that creates the diagram, then the package is not suitable and it is better to resort to plain Ti*k*Z for such a job.

The remainder of the paper is structured as follows. Section 2 explains the steps that automatize the creation of the diagram. Section 3 details the operation of auxiliary components such as the key-management interface and the library to include additional elements to the diagram. Section 4 presents some examples and Section 5 briefly discuss the media coverage of the package. Finally, Sections 6 and 7 conclude the work and acknowledge contributors respectively.

2 The core idea

This section overviews the core idea that allows to automatize the creation of diagrams from lists. First of all, one, simple macro is in charge of such operation:

```
\smartdiagram[<type>]{<list of items>}
```

where `type` is the type of diagram (please refer to Fiandrino (2012) for a complete list) and `list of items` for the items that should appear in the diagram. For example:

```
\smartdiagram[sequence diagram]{%
Select type of diagram,
Count items,
Draw diagram
}
```

creates Fig. 1, which shows the essential building blocks that works both for the animated and non-animated modes. These two modes can be enforced with different macros, the above `\smartdiagram` and `\smartdiagramanimated`. The latter only works with the Beamer class. For the sake of the diagram composition, nothing changes between the two macros.

The first argument of `\smartdiagram` that defines the type of diagram undergoes a switch-case check. Then, following Fig. 1, the first operation that is performed commonly to all the types of diagrams is to count the number of items in the list. This allows to define for the specific diagram aspects like the angle of separation between the items for circular-like diagrams or the distance between blocks in a parametric fashion. By assigning to each item an ID, then it is possible to draw the interconnections.

1. Available online at: `https://tex.stackexchange.com/q/78310/13304`

FIGURE 1: The building blocks of smartdiagram operation

The animated version of the diagram simply includes an overlay specification to the item that are progressively shown. The mechanisms that merge the capabilities of TikZ and Beamer is well described in FIANDRINO (2014).

3 Auxiliary components

3.1 The key-management interface

TikZ allows to define custom key interfaces through its pgfkeys package. Smartdiagram resorts to this principle with a twofold purpose. On the one hand, to fully customize the aspect of the diagrams in terms of colors, dimensions of the boxes, the respective distance between the various elements of the diagram. On the other hand, like explained later, it allows to create separate libraries very easily.

In the following, this subsection will expose the main aspects of the smartdiagram key-management interface:

- All the keys related to smartdiagram go under the path `/smart diagram/`. This allows on the one hand to avoid conflicts with other packages, e.g., double names, and on the other hand to make easier the development and final usage. Consider for example the interaction between pgfplots and TikZ: one has always to be careful when defining specific options to the plot (e.g., the marks or number formatting) because messing up with the key paths may lead to compilation errors.

- The different types of diagrams can share keys when appropriate or not. Besides some keys that are very generic such as those pertaining to the color definition, some diagrams such as flow charts and circular diagram share the customization of the so called *module*, i.e., the single building block of the diagram. Conversely, diagrams like the bubble or the sequence diagram have unique keys for customization. Such a mix is a precise design choice: defining specific keys for each diagram would have been cleaner, but verbose and less intuitive for the final user.

- For the sake easy the code development, smartdiagram resorts to libraries similarly to TikZ. Specifically, the code is organized into definitions (containing the keys), styles (containing the styles that customize the aspect of the diagram having as input the keys) and the commands (containing the macro for building the diagram). While these libraries (code snippet below) are loaded by default, there is one that the user can use when needed, and it is explained in the next subsection.

```
\usesmartdiagramlibrary{%
core.definitions}
\usesmartdiagramlibrary{core.styles}
\usesmartdiagramlibrary{core.commands}
```

3.2 The additions library

The main purpose of the library is to allow the user to create annotations over a smart diagram, such as include additional arrows, modules or text. Indeed, the main limitation of the approach described in Section 2 is that a diagram built automatically from a list has little flexibility in terms of customization. Quite often, a user finds itself in the need of include further elements to the graphic. However, the modification of the simple mechanism of

```
\smartdiagram[<type>]{<list of items>}
```

is very difficult. How to add text or an additional graphical elements outside the diagram and in correspondence of specific blocks? The library `additions` takes precisely care of this problem.

The library provides two macros, one called `\smartdiagramadd` and a second one called `\smartdiagramconnect`. The first one replaces the basic `\smartdiagram` because it introduces an additional argument that defines the position of the new items with respect to the original diagram. Then, the second one, is employed to customize the connections between the additional elements and those of the basic diagram.

4 Some examples

This section overviews some examples of the operation of smartdiagram.

4.1 Traditional mode

First, this subsection overviews non-animated mode examples. This exposes how much it is simple to create such diagrams.

The first example is a basic usage:

```
% A new list of colors
\smartdiagramset{set color list={
blue!50!cyan,
green!60!lime,
orange!50!red},
bubble center node color=yellow!80!red
}
```

```
\begin{center}
\smartdiagram[bubble diagram]{
Cloud Computing, SaaS, PaaS, IaaS
}
\end{center}
```

that generates the output of Fig. 2.

The next example shows how to integrate additional elements. Specifically, these components are used to exemplify with a description the meaning of the titles used in the basic diagram. Notably, one of the strength of the library is that additional elements can be of an arbitrary number and this number does not have to match exactly with the number of items in the original diagram.

```
\usesmartdiagramlibrary{additions}%
% in the preamble

\smartdiagramset{set color list={
 blue!50!cyan,
 green!60!lime,
 orange!50!red}
}% A new list of colors

% An horizontal diagram:
% - first remove the arrow from
% the last block towards the first one
% - customize the aspect of the
% additional blocks
\smartdiagramset{
back arrow disabled=true,
additions={
additional item bottom color=%
orange!60!red!30,
additional item border color=gray,
additional item shadow=drop shadow,
additional item offset=0.65cm,
additional connections disabled=false,
additional arrow line width=2pt,
additional arrow tip=to,
additional arrow color=%
orange!60!red!50,
additional arrow style={}-latex},
 }
}

\begin{center}
\smartdiagramadd[%
flow diagram:horizontal]{
IaaS, PaaS, SaaS
}{
above of module1/
{Virtual machines, servers, storage,
load balancers, network},
above of module2/
{Execution runtime, database,
web server, development tools},
above of module3/
{CRM, Email, virtual desktop,
communication, games}
}
\end{center}
```

which leads to the result depicted in Fig. 3.

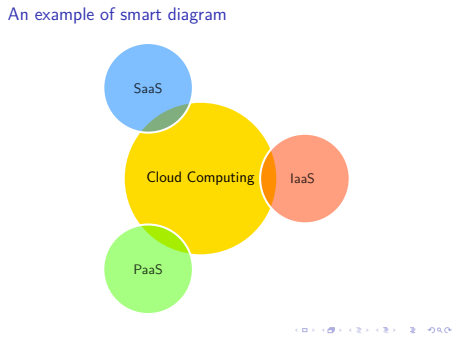


FIGURE 2: A example of bubble diagram

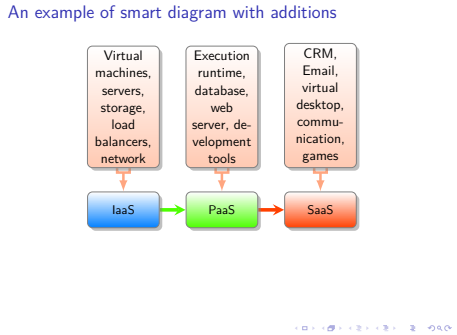


FIGURE 3: Including additional elements in the diagram

4.2 Animated mode

Another way to include descriptions is with the native `descriptive diagram`. This solution works well when the objective is to exemplify the meaning of each item in the original diagram.

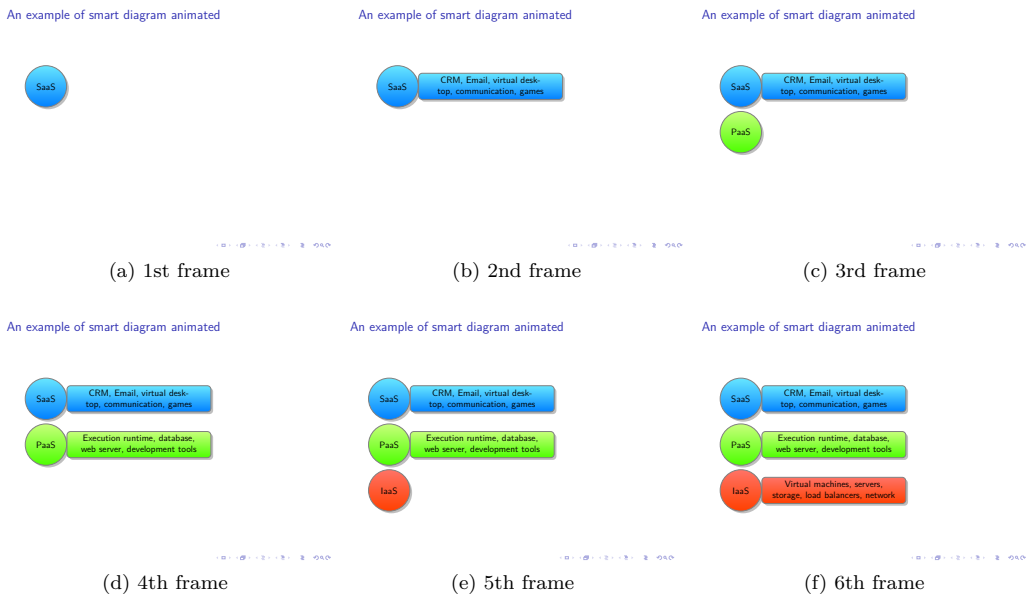
```
\smartdiagramset{set color list={
 blue!50!cyan,
 green!60!lime,
 orange!50!red}
}% A new list of colors

% A descriptive diagram
\smartdiagramanimated[%
descriptive diagram]{
{SaaS,{CRM, Email, virtual desktop,
communication, games}},
{PaaS, {Execution runtime, database,
web server, development tools}},
{IaaS, {Virtual machines, servers,
storage, load balancers, network}},
}
```

The code produces the result depicted in Fig. 4 where each subfigure corresponds to the animation steps of each frame.

5 When it got out of control

I honestly admit to be extremely proud of the evolution of the package over the years. Smartdiagram has survived pretty well the *test of time* and it is quite used within the community. Specifically, it



(a) 1st frame (b) 2nd frame (c) 3rd frame

(d) 4th frame (e) 5th frame (f) 6th frame

FIGURE 4: An example with animation

exists a tag for `smartdiagram`-related questions on TEX.stackexchange that contributed to its visibility, bug fixes and extensions.

Notably, use cases are listed in TEXample[2] and the LATEXCookbook[3]. It also exists a plugin for RMarkdown[4].

Smartdiagram was also employed in scientific publications in prestigious journals like IEEE Communication Surveys and Tutorials (mine - see CAPPONI *et al.* (2019), - and not - see SHIT *et al.* (2019)).

Also one video tutorial on YouTube talks about the package and illustrates its operation[5].

6 Conclusion

This paper has presented the smartdiagram package and has exposed the motivation for its creation, the key idea behind its operation, the rationale behind the design choices and the journey of the package along the years. The paper has also highlighted some examples of diagrams that the package can build.

7 Acknowledgements

So many people to thank for this fantastic journey. Of course, first it goes the original poster on TEX.stackexchange: good works do not come for free, but in response of specific needs. Many thanks to the numerous contributors spotting bugs (sadly, I often run late in fixing them).

2. Available online at: `http://www.texample.net/tikz/examples/feature/smartdiagram/`
3. Available online at: `http://latex-cookbook.net/articles/smart-diagrams/`
4. Available online at: `https://edpflager.com/?p=4236`
5. Available online at: `https://www.youtube.com/watch?v=1kY-1ssTk7o`

References

CAPPONI, A., C. FIANDRINO, B. KANTARCI, L. FOSCHINI, D. KLIAZOVICH and P. BOUVRY (2019). «A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities». *IEEE Communications Surveys Tutorials*, **21** (3), pp. 2419–2465.

FIANDRINO, Claudio (2012). *Smartdiagram*. `http://www.ctan.org/pkg/smartdiagram`.

— (2014). «Realizzare semplici animazioni in figure: come usare TikZ in beamer». *ArsTEXnica*, (17), pp. 18–23. `http://www.guit.sssup.it/arstexnica/`.

SHIT, R. C., S. SHARMA, D. PUTHAL, P. JAMES, B. PRADHAN, A. VAN MOORSEL, A. Y. ZOMAYA and R. RANJAN (2019). «Ubiquitous localization (ubiloc): A survey and taxonomy on device free localization for smart world». *IEEE Communications Surveys Tutorials*, pp. 1–33.

TANTAU, Till (2010). *The TikZ and PGF Packages*. `http://www.ctan.org/pkg/pgf`.

▷ Claudio Fiandrino
IMDEA Networks Institute
`claudio dot fiandrino at imdea dot org`