

The Calamus Story

After I was in 1985 switched from old 8-bit Atari on the ST series, I first developed the unpublished typography program "CAT", the similar still worked the TeX system, but my demands on usability could not reach. I could Dietmar Meyfeld convince the DMC GmbH Walluf newly established, the project of a DTP program (the term then sloshed Hot from Mac over, as well as WYSIWYG) to launch a fundamentally new. My tasks were: The user interface design and technical project management - in particular the development of all necessary techniques, algorithms and data formats. Added to this was the programming of system routines, for which I could not find a suitable programming - and I could not explain in a clear timeframe as other how they should work, so I better programmed by you.

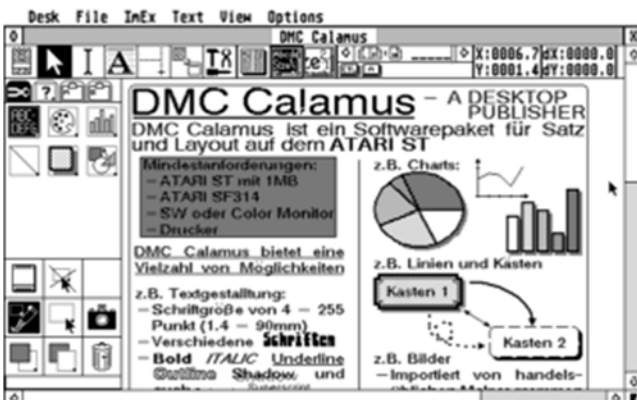
Today one would probably describe the Calamus development strategy with "agile" - redesign was one of the magic words, because the project has undergone several molts. In an interview in the ST World June 1989, once described Michel Jung (the former product manager) as follows: *"There are six full-time staff at DMC, Which what founded by Dietmar Meyfeldt at the beginning of 1987, plus 20 freelancers attached to the company. Interestingly, Mike considers the company to have a philosophical leader in Christian Griesbeck, a man who he credits with having a great creative and guiding influence on the team. There were no less than 12 names in the list and Calamus credit DMC therefore employ student programmers all over Germany, each of splat has been working on an aspect of the Calamus program. "*

In retrospect, it is still a wonder and madness that Calamus is ever arisen under the existing circumstances. Today such a small company would do something simply no longer dare you would need in the prior art maybe 20 full-time programmer for two years to to raise a similar complex project. Calamus was in reality a salaried full-time programmers and the core of a handful of students who pre-financed the project with their good will and promises towards their work. Right here but one root of dispute and decline was also simultaneously set. The underfunding, disappointed expectations, ideological narrowness and fear to be too large, Calamus have stopped at the crucial moment.

The development of the DTP program Calamus is a story with tension, intrigue, sex and crime; a story about a lot of money that has been earned, and much money was lost; a story about unfulfilled expectations to broken promises and even fraud. The most of this, dear reader, but I will omit or only suggest, that is material for a novel or a soap opera on television and not for a web page. Because Calamus is for many a success story and a beautiful memory of a brilliant product - and how this arose, I would like to snatch from oblivion and report it.

Christian Griesbeck, in May 2010

version 0.01



Screenshot Calamus V0.01

The beginning was a chicken or the egg - to convey my design, I had the many icons and their function describe. How to describe a DTP system? Best in a DTP system, which was not there but. So I solved it with the combination of images and text that I put together in a graphics program. Here I developed the operating structure, which should characterize many Atari ST programs later. Above were the normal drop-down menus. should next on the free space to the right, as was the case my program PICASO previously, future emerge explanatory help texts when driving over icons. A horizontal row of icons presented the various basic editing modes are (frame editor, text editor, etc.) - which would later be called at Calamus SL "modules". In addition, then icons for zooming, for scrolling through the pages, a display of keyboard shortcuts and the coordinates of the cursor on the page. but most important part was the great Menupalette on the left side. Above some riders (who were sometimes even bisected lack of space), including a lot of icons. The area was systematically divided - on top of a white area of the icons for the objects, eg text frames, graphics frames, etc. and framed below the icons for functions, eg, Move, Forward, etc. - this should the user, despite the set of features that provide a quick visual access.

Now the first salaried programmer of DMC was used: Harald Siegmund. He had previously been introduced with a great game for the Atari 8-bit computer "gardener" and was hired by Dietmar. Later he should be for Calamus "war effort", because he said that many items together and bore the brunt of development. But first, only the user was his job once interface in C language to get started. In the first demo version the workspace still consisted of a dummy image, only the user interface elements could be click and play. While Harald brought the User Interface for a trade to work, I took care of the writings.

font Editor



First, I developed the character set format of Calamus and a new font editor, this time a vector font editor. Editable vector graphics I already knew from "Pinball Construction Set". "GEM-Draw" - the reference graphics program on the Atari ST at the time - on the other hand was very rudimentary and could hardly serve as a model.

In character set you should next lines can also use curves - because their absence was a clear shortcoming of a PC program that I had previously looked at me as part of CAT and the simple vector fonts demonstrated. I chose the easy to be calculated Bezier curves with two control points, which were also used in PostScript. And I made a second loan from PostScript - the "Windingnumber polygons" in which the direction of rotation for overlaps decides whether an area is white or black.

For the kerning (a function in which letters may overlap to produce a better aesthetic image) I then chose, however, already a path other than PostScript, which defined only some selected pairs. Already CAT examined the letter edges to automatically generate undercuts can, for Calamus I modified this idea. The letters were divided on both sides in each 8 vertical segments defined for each, how far a different letter could zoom back there. The minimum of the eight segments of the approach of was formed, and already had a simple automatic kerning.

The editing in the Font Editor remained very rudimentary - Selecting and moving, showing, adding and deleting points and paths. Few functions facilitate the work, such as changing the direction of rotation, a fillet, a grid, guides, and a clipboard. The maximum resolution that could be achieved by zooming, were 1024x1024 - one sixteenth of the actual resolution, which should remain professional fonts reserved. A highlight was perhaps the calculator, in which one could convert using formulas the signs (or parts thereof) (eg italics). The Font Editor was, as before, CAT programmed substantially in Pascal. To digitize existing fonts on a screen grave function easier, I had programmed him as an accessory.

The time-critical functions for rendering letters from the vector description in a raster graphics I wrote in assembler. That was quite a complex task, because at first the description of "Move", "Line", "Curve" and "End" in polygons is converted from straight lines, this split vertically and sorted, and then turn abgehangelt for output line by line and issued with horizontal lines. As icing on the cake, the whole thing was built with tricky Integerarithmetik, because that was the quickest, what the blazes 68000. This routine was also a first base for the text output in Calamus. There was but overall everything again much more complicated, because the issue had even faster, and different fonts and formats are considered.

The vector fonts were a main feature in the Calamus all other DTP programs would be for years ahead. While we were working on Calamus, Adobe failed because establishing Display PostScript on the market. WYSIWYG was on the Mac for a long time a cheap grobpixelige preview - half a flying blind. In Calamus you really saw, however everything on the screen, what you got - because printers, imagesetters as well as the screen output using the exact same routines and fonts. You could easily to any printer resolution and even further back, without a vector element as Scripture could be pixelated. To quote the former Page Author Matthias Schrader: *"While one saw on the PageMaker headlines in blocks optics (Adobe moved until four years later the ATM out), could with Calamus end of 1986 already on the screen 1: 1 in the imagesetter 2540 dpi work. It haute us off our socks. Adobe and Steven Jobs attacked the concept a little later under the heading Display PostScript on - but it made war under NeXTStep never performant"* (http://www.fischmarkt.de/2006/05/schuelerzeitung_goes_dtp.html)

The child gets a name

Calamus

The name and the first logo of the product I had soon found "Calamus" a reed that was formerly used for writing. At first there was a little doubt whether Calamus and calamities are not too close together. Then there were problems with the protection of the name - the Patent Office said that it is not possible to protect the name - Calamus is the name of a plant and therefore the word is not protectable! So was Alfred Smeets (would later also design some fonts for Calamus) a new logo designed - the one with the famous Calamus spring - entered and only as a picture / word mark. About a year later Linotype had the better Pate lawyers. They made sure that them (previously DMC denied) rights to the word mark equal for all possible areas conceded the Patent Office for the name of a new font. The rights then had to buy expensive DMC - for far more money than for some important system component. Blessing in disguise - Linotype was interested to come with DMC into business to sell securities - otherwise a new name would probably be even more expensive to be established.

Memory management

An important part of the Calamus required, but the Atari operating system "TOS" not lent, an effective memory was manager. This is another part of the Calamus team was tasked developed it independently. Klaus Garms and Pierre Hansen were the core of the "Darmstadt group", they had worked together, and a driver for the Atari laser printer developed which simulated an Epson FX80 on them. The problem was that an A4 page with 300 DPI MB needed to store more than many Ataris had ever at main memory. So the page on-the-fly paved the

tricky driver during the current printing häppchenweise just in time to. This program was then distributed by DMC.

On the Memory Manager also some on-the-fly were asked requirements, and he grew up with these requirements. A Calamus document consisted of many individual objects that would be stored in each case in a linear memory area, as well as temporary storage, which was required for many functions. The objects could change their size, be deleted or more. This was achieved by a handle system, so the objects could be moved in memory, if memory has been released to re-establish large contiguous areas. However, a handle could also be blocked in order to prevent displacement of a storage area, the one needed at the moment. Later could even the hard disk will be used as virtual memory.

document format

What does a Calamus document internally from? How are the individual objects constructed? How is the text format, etc.? - These are the questions I had to answer. I was able to rely in part on the experiences with my preliminary CAT, but decided the text into a space-saving binary format.

For Harald be peeled so slowly out that he probably had to make more than the programming of the user interface, for his next task was a frame editor, and you could do something right for the first time in Calamus. First, the GEM functions for drawing lines and surfaces were for demonstration purposes still used, but it was clear that this would not be suitable and powerful enough for the future.

Vector output

Since I had already written an output routine for vector drawing I made this greyscales via a bit pattern. However, the greater challenge was to draw thick lines. Sure, you could as GEM with a brush function along draw the lines, but it was slow and resulted in moderate results. My model was the powerful line function of PostScript, there you could determine whether and how the lines should be dashed whether corners should be pointed, round or flat and as the ends should look next to the width. To realize that there was only a meaningful way, you had to the outer edge (outline) of the corresponding line are calculated as vector graphics - so to speak along a polyline to both sides build boxes. Thanks to the Windingnumber polygons I had to at least do not care about self-overlap, and the modified output routine for vector characters could spend the calculated line polygons. All this required some trigonometry and solving equations in Assembler. Not a very fun task, but I managed. I do not think that anyone has ever understood how the routine works. When transferring to the PC then the machine code was blind translated into C and it worked.

Raster image output

I would time barely manage to program all output functions myself in assembler, I practiced last in delegating and transferred the routine for the scaled output of raster images Jochen Mickel. His father worked at Motorola even, so that the 68000 was almost in the family. I told him what we needed and as an integer version (Bresenham) a Digital Differential Analyzer works - with two of them (one for each axis) could in my opinion quickly and easily resize images. We needed an output routine for black and white images and grayscale images, while we experimented beside a dot grid with grids by a Floyd-Steinberg error diffusion. Everything else I left Jochen, who coordinated with Harald to integrate the routines in Calamus. In later versions of Calamus routines but were again replaced by newly programmed.

text output

The hardest chunks in Calamus was certainly the text output, which had to go through in the course of time several molts to become better and better performance. It is certainly the greatest marvel in Calamus, the implemented Harald. How it would work with the kerning, was easily explained and programmed a simple text output routine soon. It quickly became clear that you can watch the text output, if it would stick to a pure vector output - a brilliant idea came here by Klaus, who said that one in computer science by storing and reusing intermediate results (caching) speed things can. Also PostScript used for faster outputting character a cache, but which was built for a complete character set and so when unused characters unnecessarily swallowed time and space.

In Calamus caching was henceforth a charm. If a mark for the first time in a particular font, requires a certain style and a certain size, it was initially stored in the cache and then brought it with a fast bitmap copy function on the screen. Here, however, always had the limited memory of the Atari into account, so be longer no longer used things also disappear from the cache and can also be decided whether it is not each meaningful character but directly, by means of the vector output, on the screen bring to.

Not only the characters ended up in a cache, but also information about the transition and the position of each character within a line. For this, the position has been normalized to be independent of the zoom factor - which also sparked a common problem that had other programs that calculated the text set in the low screen resolution and then looked completely different with the high resolution of the printer. Calamus let the cold and so was the first DTP program with real WYSIWYG.

To be continued ..