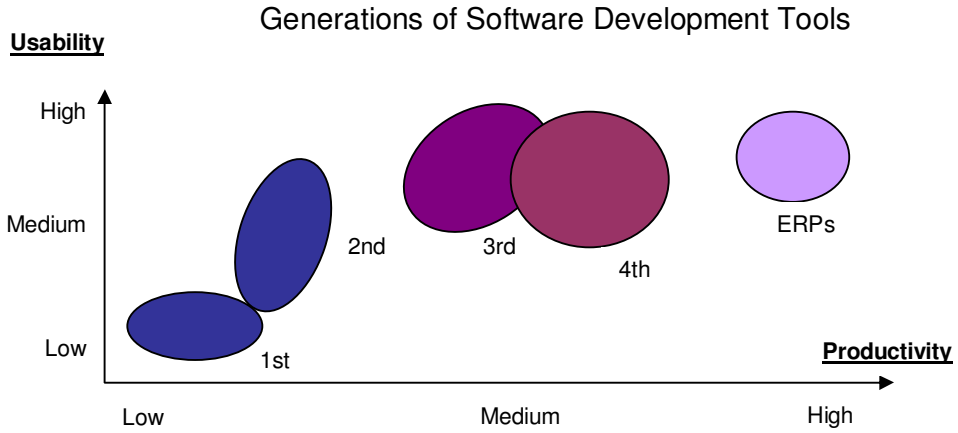
Software Simulations: The drive behind 4 generations of tools

During the nineties, the software development industry underwent a radical paradigm shift, within a matter of a few years, from First Generation of development tools (Assembly language, COBOL), to the Second Generation (Pascal, C), a Third one (Visual Basic, Delphi, PowerBuilder, Centura), and even a Fourth one (Forte, NatStar) which was eventually overtaken by the ERP wave before it could established itself.

The main drive behind this evolution was productivity and maintainability of the code produced, and usability (interactivity) of the application (user interface). Cost (productivity) and maintainability were also the main factors behind ERP adoption.



In the last five years, the development of the Software Simulation tools within the eLearning industry has undergone a very similar path, driven by similar variables: productivity, maintainability, interactivity.

A first generation of tools borrowed from another market…

In the mid 90s, before the hype of eLearning, most training on large IT software environments were taking place in classical classrooms, with the full infrastructure, a training database (sandbox), and a significant challenge in organization and technology. At the same time, the first embryo of software simulation were appearing in the form of videos (Lotus ScreenCam), and PowerPoint animations. The near-zero level of interactivity of these solutions gave them a very limited use within the Training Organization, but they still held their place in a presentation / information phase. They also helped prepare the space for a second generation of tools which would bring in more interactivity and more pedagogy.

…opens the way for a second one, with increased value for money…

The second generation represented a true paradigm shift from the first. It is essentially a Point & Click solution, associated to basic Data Entry capabilities, the whole solution being based on screen captures of the software to simulate. This great enhancement comes with an associated burden: a longer development process, and induced lack of maintainability. Still, the gain in interactivity and pedagogy meant that this solution had a far better place in a Training Organization, and positioned itself as a very good alternative for software demonstration. The capabilities of this second generation of tools also helped the birth of the eLearning concept and industry. It also certainly carried too much on its shoulder, and led to major gaps between expectation and reality, hence creating a space for Blended Learning.

…but too high unrealized expectations call for a repositioning and a third generation…

So software training had to become blended. A third generation of tools began to enter this space, addressing some of the key issues associated with the previous generation, and facilitate Blended Learning.

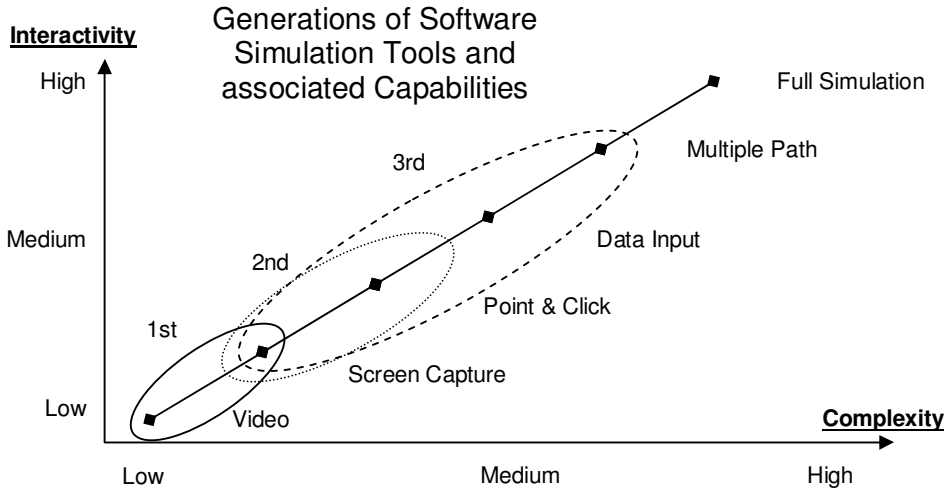
This generation mainly builds on the second generation, to add further productivity, interactivity, and diversity in the training material delivered. Where the previous tools required hot-spot and edit fields manual insertion on the screen copies, the new ones generate most of these graphical objects automatically. Let's say that it cuts the development time, thus rendering the training solution itself a bit more maintainable. It is also generally possible to manually insert multiple path capabilities into the simulation, to render the look & feel closer to the real software, therefore helping the trainee have a better eLearning experience.

The main purpose of this new generation is to overcome the demonstration mode, and provide richer training content. The third generation of tools have therefore invented and expended their own space, with new essential concepts such as the famous "See-it", "Try-it", "Do-it" modes, which have been marketed so strongly they are now a must-have. The whole idea is to provide several training resources from a single source. Hence the delivery of different training materials for different needs: demonstration, exercise, evaluation, transactional documentation, EPSS, etc…

…based on same technological foundations, and thus limiting the scope for progression.

In some respects, the third generation of simulation tools has similarities with the third generation of development tools. You get more for your money: increased productivity, enhanced interactivity, and more diversity.

However, as an evolution from the second generation, it is not a paradigm shift, as had been witnessed between first and second generation of simulation tools. Simulations produced are still based on screen captures, and in fact, the final gain in interactivity is strictly due to the gain in productivity: you can add more interactivity because it is easier to develop, but you are still limited by the "Screen shot Boundary". Also, the same screen copy technology brings a limit in the maintainability of the developed material, as most maintenance will imply screen shot recaptures, which is unsustainable in large corporate projects.



A new generation based on Software Cloning is emerging…

If we are to bring a new generation of tools, it needs to go beyond the barrier of the screen copy. This new space is being filled by tools which virtually clone software. Software cloning allows for high interactivity, high productivity, and high maintainability. Based on graphical object recognition and emulation, it brings a real paradigm shift and promises to deliver to the trainees a true alternative to the sandbox and training databases.

…to bring trainee experience and project maintainability to the next level…

As software itself is ever increasing in complexity, training environments based around Software Simulations are becoming essential training systems. However, for these Software Simulations to have a permanent and strategic place in the training organization, two key groups have to see their requirements addressed:

- the trainee's requirements: interactivity, pedagogy, diversity of the resource
- the project's requirements: productivity, maintainability.

To date, diversity of the resources and pedagogy have mostly been successfully addressed by the third generation of tools. Full interactivity of simulations will be provided by the new cloning technology.

Productivity has always been a main drive within this industry, and there are frequent challenges where one can see software publishers compete to create full training environments based on simulations, in less than 15 minutes. Therefore, producing material quickly has never been seen as a differentiating factor, as most products across the full range of generation tools have roughly the same speed ability.

The real challenge is maintainability, as in the case of most projects, the software evolves every quarter, the training environment need to follow suit (understand your 300+ simulations, demonstrations, exercises, evaluations, documentations, EPSS, etc… that have been created for version one of the software, now need to be revised).

…thanks to new capabilities based on a technology paradigm shift.

Fourth generation products address this issue of maintainability with the innate advantage of object recognition, versus image made of pixels for all previous generations. This additional information (the capture of both object properties and user actions) allows new capabilities:

- Simulation self testing (auto detect consequences of software upgrades in simulations already created)
- Automatic simulation recapture (auto maintenance of training material)
- Automatic simulation localization (via dictionary filters)
- Repository of objects (a single modification has a global impact on all simulations created)

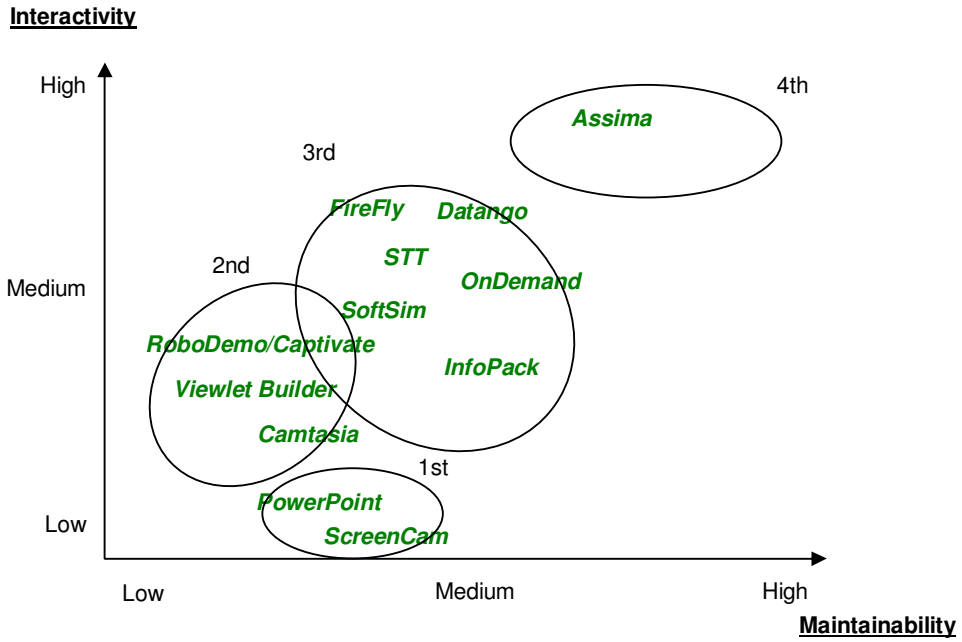
On large scale projects, these functionalities may make the difference between "can" or "cannot" do the project, taking in consideration quality, cost, and project life cycle.

What for the future of Software Simulation Tools?

The development tools market has matured during the last decade to leave space for the third generation of tools (for small scale projects), and the ERPs (for large scale projects). The market share of Assembly Language or Pascal is reduced to the bare point of technical necessities.

It is likely that the Software Simulation Tools niche-market will also structure itself rapidly. The First Generation was merely there to give birth to a new market. The technological difference between the Second and Third are not strong enough to justify the continuation of such large difference in cost of product ownership.

Generations of Software Simulation Tools



The future will certainly retain only three types of solutions:

- Third generation tools for the price of second generation tools. These will be used on most small or medium scale projects (less than 100 simulations, not critical to core business, or with used in a Blended Learning context, with sandbox environment). The probability is that Second Generation tools will catch up with the functionalities of the Third Generation (RoboDemo becoming Captivate, and now including more and more functionalities). The maximum capabilities being defined by the limits imposed by screen shot captures, it is just a matter of time before the cost for the full functionalities decreases dramatically.

- Fourth generation tools for most large scale customized and critical projects. Software cloning will have the edge in a foreseeable future on these critical projects because they offer the right level of interactivity and maintainability.

- Ready made training packages for all widespread standard systems (OFFICE, ERPs, CRMs, etc…). Most of these packages will themselves be generally built with third and/or fourth generation tools, depending on the size and criticality of the underlying product.