Cocoalytics

Sebastian Fichtner

February 19, 2019

Contents

1 Product Idea

1.1 Original Note (The Napkin)

tool das grafisch das ood in einem softwareprojekt darstellt und zum teil refactoring untersttzt ....quellcode nach imports durchsuchen.... mglichst baumdarstellung der abhngigkeiten finden (main = wurzel) ... visualisieren: zyklen, mvc, daten die privatisiert werden knnen, indirekte abhngigkeiten durch fehlende imports ..... Marktlcke entstanden: swift files haben weniger explizite Abhngigkeiten weil es innerhalb eines Moduls (targets) keine imports gibt, es werden keine files sondern meist nur ganze frameworks importiert, man knnte sich auf swift spezialisieren und auerdem: das tool als online tool das den code in git repos darstellen kann tool bruchte evtl. zugansdaten bei privaten repos dann knnte jeder diesen online service sofort nutzen.

Future name: Codeface

1.2 Value Proposition

- Overview over a code base, like a map provides overview over a territory
  - seeing general architecture and relationships between entities at a glance
  - getting back into your own code after short and long breaks
  - getting into other people's code and legacy code
  - maintaining context during development and avoid getting lost in details
- Visualization
  - more balance and effectiveness in leveraging cognitive capacities by complementing textual with visual artefacts
  - easier communication of abstract properties of the code by having explicit artefacts to refer to
- Automated diagrams
  - strong support for domain-driven design as diagrams and code are always congruent. you change the design/diagram by changing the code.
  - Relieve developers of having to create and maintain class diagrams
- Objective quality standards
  - Objective Insights into how different parts of the code compare with regards to different quality metrics

  - Objective measures for what parts of the code are most essential to unit test based on ROI of testing. (As opposed to the ideological black and white (all or nothing) stance that many developers take so they don't have to navigate through a grey area)
  - Objective data-based suggestions on how to improve the code
  - Better communication and more consensus on quality standards

- Real time monitoring of quality
  - More flow during development through immediate feedback
  - Increased awareness of how code rots or can be improved over time
  - Knowledge in action, learning by doing and seeing

1.3 Unfair Advantage

With this project, I can leverage 5 distinct areas of my expertise or passions:

1. Software Development and UX
2. Creativity-Support Tools (master-thesis, flow, creative process, psychology)
3. Data Mining & Visualization (in particular concerning graph and text data, my master in Information Engineering covers exactly that)
4. Clean Code & Architecture (specific passion of mine that not all developers share, also developed an original idea here...)
5. Writing & Teaching (as part of the product and its marketing)

In addition, I know people who research in the field of data- and even source code visualization.

2 Market Analysis

2.1 Market Size

- 2014: 320k paid Apple developer programs
- 2014 June: 9 million registered Apple developers, 47% increase from 2013
- 2014: about 5.5 "professional" mobile developers worldwide (mobile not including Mac OS)
- let's say there are 10million active XCode users... we get one in 10000 to pay 5 bucks... that's 5000 bucks which is NOTHING!

2.2 Ways to scale the income up

- branch out to other object oriented programming languages that are used on mac: Java, C++, Ruby ...
- subscription-based payment
- emphasize the monitoring aspect: let non-technical staff monitor development (include management)
- emphasize the educational aspect (include people who have source code but just want to learn about clean coding)

2.3 Niche & Competitors

1. Category: Analytics tools for software development
   (a) codalytics.com (workforce management)
   (b) crashlytics.com
   (c) localytics.com
   (d) google.com/analytics
   (e) klocwork.com (non-visual code analytics)
   (f) https://codeclimate.com/quality/ (platform for code analytics plugins, very few plugins, not really visual)
   (g) https://www.codemetrics.com
2. Market: Visual Code Analytics
   ...no tools that are ONLY available as IDE plugins, since their (visual) capabilities are limited
   (a) lattix.com (no Swift support)
   (b) semmle.com (cocoa support unlikely, aimed at executives, managers and developers)
   (c) grammatech.com (cocoa support unlikely, security oriented)
   (d) ndepend.com
   (e) imagix.com
3. Niche: Compatible with Mac OS or cloud-based (developing **on** Apple platforms, developers with taste, creative developers)
   (a) scitools.com (no Swift support, no simple download, no price transparancy, technical and complicated)
4. Micro Niche A: Supporting Objective-c and Swift (developing **for** Apple platforms, Cocoa projects, ca. 15 million registered Apple developers, even greater emphasis on simplicity, fun, creativity and aesthetics)

   (a) seerene.com (under construction, raised their capital in 12/2015, support for Objective-c and Swift possible but unclear and unlikely to come in 2016, heavily aimed at big companies)

5. Micro Niche B: Servicing "small" customers through simple app download (simplicity of consumer product, for freelancers, solo developers, startups, small companies)

   (a) structure101.com (no support for Swift)

2.4 The Five Closest Competitors

1. structure101.com

   (a) What they sell: An app that provides static code analysis to developers
   (b) They require to fill out a form including email address to download the free 30 trial version
   (c) They provide product demos and talks on their Vimeo channel
   (d) They offer free licenses for academic-, classroom- and open source projects
   (e) They offer a 15 day "no questions asked" refund policy
   (f) They provide a Blog, but it's not very up to date
   (g) They maintain a twitter profile
   (h) Their newsletter requires filling out their contact form
   (i) Their trial request is still the best: It only requires name, email and company, and it lets customers download the app immediately.

2. seerene.com

   (a) What they sell: An app that provides visual code analytics to CIOs and stakeholders
   (b) They have an impressive beautiful landing page
   (c) They maintain a LinkedIn profile where they post updates that link to interesting related blog posts all over the web
   (d) They maintain a twitter profile where they post the same links as on LinkedIn, among other tweets
   (e) ... and a Product/Service page on Facebook with the same updates
   (f) Their newsletter requires filling out their contact form and sending a message
   (g) the 2 problem here: 1) they've just been founded and are in the seeding phase, putting the millions they raised to use. it'll take a while until they provide a product. 2) they aim specifically at big players who have CIOs and can afford a very complex integrated solution.

(h) Q: When will there be a trial version of Seerene?
A: Never. Seerene is no stand-alone frontend product, but a SaaS solution connected to business software that is being analysed in real time.

(i) Q: On which platforms will Seerene run?
A: Seerene already runs for some time on our Server-Farm, that is being provided by a Banking-Level Security Service Provider. The Software that connects the business applications with the Seerene-Backend runs on all common operating systems - but as I said, Seerene ist no Frontend Product, at which your question probably aimed.

(j) Q: Which Programming languages will Seerene be able to analyse? Also Swift und Objective-c?
LONG ANSWER:
Seerene currently supports ca. 60 languages i.e. all languages that are commonly used in medium und large sized companies. Currently, about one language is added every month, which so to speak is a regular byproduct of our client project activities. Swift and Objective-c currently not, since our focus is on larger and very large business software, where both languages play almost no role. But we also work together with several famous '.com' companies, that require more and more to integrate languages that are unusual from a traditional developer perspective, which we, as i said, also currently do with one additional language every month step by step.

According to your questions and you company-website your focus is presumably on client projects of iOS Apps, which are by their nature probably relatively "small" and behind which is a rather small Resource-Budget (Manpower/Cost). I don't mean that degrading in any way - but the point is, that the central value proposition of Seerene is to support clients to radically reveal hidden costs, risks and complexities in rather large applications, so that these can be reduced systematically step by step.

That means that Seerene generally requires a certain 'flywheel mass', to be applied meaningfully. In situations where, for example, a small application is being delivered to a customer once as a finished work, applying seerene is rather not reasonable and also commercially not viable, and in such a situation, the client will mostly have no motivation to let a small and budgetwise not very relevant application be analysed continuously in real time.

I intentionally sketch out the basic guardrails here, so that you can accurately assess what we're all about. Should there be meaningful connections from your point of view, I'm happy to hear from you.

3. scitools.com

(a) What they sell: Apps that provide static code analysis to developers

(b) They maintain a Blog that is technical and focused on their specific product (no general interesting posts)

(c) They responded to my trial request by emailing me a link to the download page

4. lattix.com

   (a) What they sell: What they sell: An app that provides static code analysis to developers

   (b) They maintain a Youtube channel with product demonstrations, a Blog and a twitter profile

   (c) Their "customer portal" requires a login

   (d) They offer consulting and training for the use of their product (which speaks to how little intuitive and simple those products are) ... no price transparency here either

   (e) They responded to my trial request several days later by email and suggested we have a phone call: "To discuss your needs and the evaluation process please let us know when you have time for a short call." ... clearly ONLY targeted at classic B2B custom business-software...

5. semmle.com

   (a) What they sell: A suite of apps that provide static code analysis to executives, managers and developers

   (b) They have a LinkedIn profile without any updates, a newsletter and nothing else

Only Semmle offers an independent newsletter that is not tied to the request form. Seerene and Structure101 offer newsletters. In general, newsletters seem to be out of date. Competitors prefer to spread the little news they have through twitter, facebook, linkedin etc...

Twitter profiles and Blogs are always half informative and half promotional. They try to weave their products into the public conversation.

Some have more online profiles than listed here (twitter, facebook ...) but those profiles aren't worth mentioning (too little content and traffic)

Structure101 is a bit better, but in general, the only way to get infos on what exactly competitors offer or where and when to get the app and at which price is a contact form on their website. There is no explicit download option. Customers need to fill out a form to even get the trial version which they receive a few days later. That is annoying for customers and wouldn't work if you had many small customers instead of a few big players cause it doesn't scale.

Most competitor's websites lack transparency they don't explicitly state:

1. on what platforms their tool can run

2. what programming languages their tool can actually analyse

3. what additional tools or steps are required until a project can actually be analysed

4. what the tool costs

2.5 USPs of Cocoalytics

Except for structure101.com, all competitors aim at big companies and classic B2B. They offer no trial app downloads but require to fill out complex forms to contact them. Many don't even display pricing information on their website and offer no free value beyond limited evaluation periods (ridiculous selling funnel) or a few blog posts. Their products are very technical oriented, ugly and have no educational aspect. Most of all, they don't support Cocoa projects (Objective-c and Swift).

Even structure101 tends in that direction: its personal licence costs 395$. The download requires filling out a form. The app is not available in the App Store and is as technical and ugly as the other products.

There are only few products out there that are remotely similar to Cocoalytics. Cocoalytics easily sets itself apart:

1. **Mac**: Developers using Macs have almost no products to choose from. They are underserved because competitors aim at B2B and developers of critical software which mostly is developed (and runs) on other platforms. Or so it has been. As mobile software (and thereby iOS apps) become more and more professional, powerful and critical, developers with Macs need more code analytics.

   Key point to understand: In general, software for apple platforms can **only** be developed on Macs, while software for other platforms can **also** be developed on Macs, although native Windows software is **mostly** being developed on Windows.

2. **Cocoa**: Support of Objective-c and Swift, Specialization on Mac OS, iOS and tvOS (developer needs, frameworks...). Btw: Swift is now open source and will eventually conquer all platforms. It's the future.

3. **Simplicity** (low entry threshold): Just download, open, select project directory, done. Not required: git, login, host application, request form, additional software, console commands, configuration, etc.

4. **Small Customers**: Serving solo developers, educators, consultants, startups, small companies, individual developers

5. **Availability**: There is not a single remotely comparable app among the top 180 developer tools in the app store. Not even any static code analyzer.

6. **Visualization**: Non-technical people (like SCRUM masters) can use and benefit from the app. Focus on feedback, not editing. No interference with established workflows. Interface and visualizations are not just functional but also beautiful, engaging and promoting focus and flow.

7. **Architecture**: Convey information about higher-level architecture and design. Utilize en.wikipedia.org/wiki/Software_package_metrics and incorporate en.wikipedia.org/wiki/Package_principles.

8. **Quality-oriented**: Convey not just structure or explicit defects but also general code quality metrics to avoid software rot in the first place. No focus on team controlling or git activities.

9. **Action-oriented**: Main intent of the output is to suggest how the code could be improved. No overwhelming data dump. No useless technicalities. No pointless fancy eye candy.

10. **Learning**: Educate about the quality metrics, principles and ideas behind Cocoalytics. Let user explore those background infos on demand.

11. **Real-time** feedback: Immediate monitoring and code exploration during programming. No upload to git repo required. No email digest BS.

12. **Pricing**: Competitive price and price transparency. Most competitors don't even name a price.

3 Customer Analysis

3.1 Search Term Prediction

- (source) code visualization (tools) → tools, c, php, java, python, eclipse, c#, free, video, javascript, open source, c++, objectice-c, visual studio
- software visualization (tools) → for debugging, tools, architecture, development, free, porting them to the web, open source
- (source) code quality → tools, metrics, java, analysis, sonar, measures, assurance, assessment, measurement, standards, check, criteria
- software quality → assurance, metrics, assurance jobs, assurance engineer, assurance training, analyst, engineer salary, assurance salary
- software architecture → diagram, patterns, in practice, document, interview questions, design, books, document template, diagram tool
- software architecture diagram → tool, example, visio, types, template, tool mac, software, symbols, visio template, powerpoint
- source code analysis → tools, java, open source, laboritory, php, free

- software quality metrics → dashboard, examples, tools, overview, methodology, formulas
- (source) code quality metrics → sonar, c#, java, jenkins, tools, visual studio, eclipse, python, measurement, check, open source

The search term prediction didn't reveal much new information except a number of programming languages for which people want to have quality analytics the most: java, php, c# and python came up several times.

3.2 Communication and Problems

- structure101.com: blog and https://vimeo.com/structure101, here are some customer tweets:
  - "qotd: why does it always have to get worse before it gets better? (refactoring with @structure101, tools is great, but my design is awful)"
  - "... I really like structure101 (although its not a build plugin)"
  - "i know one clean open source project, the Spring framework. And they used tools like Sonargraph or Structure101."
  - "structure101 is the tool to help you splitting tangled monolith into a bunch of micorservices"
  - "cloc, gource, structure101, sonarqube are the tools I use to familiarize myself with unfamiliar java (and non-java) code bases"
  - "Tool to analyze architecture entropy. structure101"
- stackoverflow.com (technical question-answer platform): Most people want to know what code visualization tools exist at all and whether they do what they need.
  - "I'm trying to get a grip on a large code base that has hundreds and hundreds of classes and a large inheritance hierarchy. I want to be able to see the "main veins" of the inheritance hierarchy at a glance - not all the "peripheral" classes that only do some very specific / specialized thing."
  - "Basically I want tools which generate source code visualization like: function call graph, dependency graph ..."
  - "Is there code visualizer available (preferred as eclipse plugin)?"
  - "is there a code visualization plugin for eclipse? ... it's too expensive however so is there a open source version?"
  - "Is there a code visualization website for Racket programs (for novice WeScheme users) similar to what is available at Online Python Tutor? Needless to say, it would provide a great self-teaching or learning tool."

- "... intention being to enable a user to manually track interconnectivity between different parts of their code for the purposes of implementing specific features. In particular, a user could pull out a window into their code and associate it with a completely different part of their code, seeing lots of independently controllable code snippets (and any relevant notes) linked together. The purpose of this tool was to enable someone working on a chunk of code that crossed multiple functions (or separate modules) to be able to more cleanly figure out what was going on and to see more of the relevant code at once"
- "coming across a lot of reasonably large, complicated codebases at work recently which I've been asked to either review or refactor or both. This can be extremely time consuming when the code is highly concurrent, makes heavy use of templates (particularly static polymorphism) and has logic that depends on callbacks/signals/condition variables/etc.

  Are there any good visualization tools for C++ period, and of those are there any that actually play well with "advanced" C++ features? Anything would probably be better than my approach now, which is basically pen+paper or stepping through the debugger. The debugger method can be good for following a particular code path, but isn't great for seeing the big picture you really need when doing serious refactoring."

The problem mostly is finding and quickly evaluating tools that provide overview of complex code bases and help with refactoring and producing clean architecture and clean code i.e. software quality.

Cocoalytics adresses that by providing a low entry threshold (good availability, educational aspects, consumer market vibe and ... well, all the USPs listed previously.

3.3 Customer Survey

http://bit.ly/1UVdZBF

1. What metrics do you use to estimate the code quality and maintainability of a software project? Do your colleagues agree on these metrics?
2. How do you maintain overview over complex code bases, in order to keep the architecture and design of your code clean and maintainable over long time periods?
3. How you and your team communicate the architecture, design and quality of a code base in order to improve it?
4. For what software, books or other products have you (or your company) ever paid to help you produce clean maintainable code? What did you hate or love about them?

5. What techniques/tools (manual, automated) have you ever used to generate diagrams from source code. What did you hate or love about these techniques/tools?

4 Product Ecosystem

4.1 Free Products

1. Cocoalytics Lite: Demonstrates graphics, look and feel. Is fun to use. Playful. Makes development more exciting. Makes itself as well as architecture, design and quality of code a subject of discussion in teams. Supports as many programming languages as possible!

2. Blog: Posts on software development, in particular on architecture, design, quality, metrics, visualization etc... (already got material) ... possibly video logs... integrate the posts into something whole like a book/wiki.

3. Some open source code for the community via github and cocoapods.org

4. Some kind of forum or network for a community of cocoa craftsmen. Relate to cocoaheads.org? possibly build local group (already got cocoaheads-bodensee.de for that)

4.2 Products for Prospects

1. Cocoalytics Pro. It has more: metrics, graphical mappings, visualization types, customization options, saving project files... pay once for the app.

2. By writing, researching and testing for the blog and for cocoalytics, I gather material for talks/videos/webinars and a book i.e. some form of educational product.

4.3 Core Product

1. Cocoalytics Master: login, build integration, web dashboard, github access, IDE plugins, customer support. subscription payment.

2. Individual consulting and workshops

3. Add custom features to Cocoalytics (for individual clients) thereby turning it into business software and a B2B operation

4.4 Success Path

What Customers Achieve clarity, perspective, fun, overview, understanding, flow, productivity, clean code, maintainability, momentum, confidence, communication, accountability, insights, knowledge, pleasure, code quality, clean

architecture, clean OOD, pride, control, objectivity, agility, flexibility, motivation, energy

10 words: Produce better software quicker, be a proud craftsman, have fun!

Cocoalytics Pro (for prospects) Price: starting with 5$, increasing up to 90$ as product value increases with new features

1. **Awareness**: Customer becomes aware of Cocoalytics Pro through online media or Cocoalytics Lite
2. **Download**: Customer downloads the app from the App Store or from cocoalytics.com
3. **Setup**: Customer selects an existing project directory from within the app
4. **Explore**: Customer explores his code through an interactive visualization
5. **Improve**: Customer improves his code in his familiar IDE of choice (most likely XCode)
6. **Learn**: Customer sees how the code changed in terms of architecture, design and quality. He learns more about the theoretical background through the app.
7. **Assurance**: Customer feels in control and is confident in his ability to produce high quality code, design and architecture. He understands and oversees his own code better and in a new way. The additional layer of feedback pleases and leverages his visual sense and enables him to work longer.
8. **Communicate**: Customer shows his code to others through the app. He starts conversations on architecture, design and quality of his team's codebase, instills pride into them and encourages them to use the app as well.

5 Product Pitch

5.1 Pitch for LRBA application

Millions of software developers try to create clean software architecture while staring at code, juggling tons of little infos in their heads.

Forget the money that this insanity costs. Their SOULS are depleted!

I will create a service (cocoalytics.com) that provides meaningful interactive visualizations of program code. Developers of Apple software (cocoa projects) will be able to make sense of the whole picture. They will see and shape the "building" instead of a list of one million bricks.

They will think about code in more semantic terms, develop pride in their work and communicate it better to others. This will make software development more productive, more social and more fun!

5.2 Iteration 2

Cocoalytics is the tool that introduces visual feedback to software development. It provides you with meaningful interactive visualizations of Objective-c and Swift code. As a developer you attain overview and insight on the architecture and quality of your project, just by looking at its graphical representation.

You and your team will be able to monitor in real time how your programming effects the design and quality of your code. When will be able to talk about implicit qualities of your code by referring to their tangible representation. This will change the way you think about code and will bring your whole development process to a new level where it is more productive, more social and more fun!

If you want to experience more flow in programming, become a better software craftsman and upgrade your team's agility and drive, we offer you to pioneer a new way of software development. Be the first to try the Cocoalytics experience and shape this exciting product with your feedback!

5.3 Iteration 3

Cocoalytics is the tool that introduces visual feedback to software development. It provides meaningful interactive visualizations of Objective-c and Swift code, so you attain overview and insight into your code base, by just looking at clean graphics.

You and your team will be able to monitor in real time how your work effects the implicit architecture and quality of your code. This will change the way you communicate and think about software and raise your whole process to a new level in terms of agility, craftsmanship and fun.

If you want to experience more flow and team spirit, we offer you to pioneer a new way of software development. By participating in our user tests you can taste the Cocoalytics mindset and shape an exciting product with your feedback!

5.4 Iteration 4

Cocoalytics is the app that empowers software developers as well as their managers and clients by providing visual maps and analytics of software quality.

I started coding more than 20 years ago and worked as a software engineer for different companies, and I learned that managers and clients often fail to consider and understand the internal quality of software. And even the best software craftsmen struggle to apply their knowledge to real projects and to see the high-level architecture implicit in code.

Cocoalytics provides meaningful real-time visualizations of source code, so that everyone can communicate the architecture and monitor the quality of software, by just exploring interactive graphics.

And that also allows you to share objective quality standards, communicate visually, experience flow and establish accountability. You can shape the cocoalytics app with your feedback and ideas. Visit cocoalytics.com for more!

6 Notes on Raising Capital

* venture capital ist kein Kredit! der investor trgt ein erhebliches Risiko mit. das heit man ist nicht so wahnsinnig unter druck alles zurckzuzahlen, wenn man erst mal einen investor hat. die Herausforderung ist eher einen zu berzeugen... dafr mte man sich auf eine Idee konzentrieren... wenn es mit der nicht klappt die nchste, aber prinzipiell eins nach dem anderen. Fr venture capital muss man eine kapitalgesellschaft sein (z-bsp. UG oder GmbH)

* Startkapital kann durchaus sinnvoll sein, denn um in absehbarer zeit eine wertvolle software zu schreiben brauch ich zeit und evtl. zustzliche Leute.

* Grnder-kredite oder lieber langsam wachsen? Normaler Kredit auch mglich