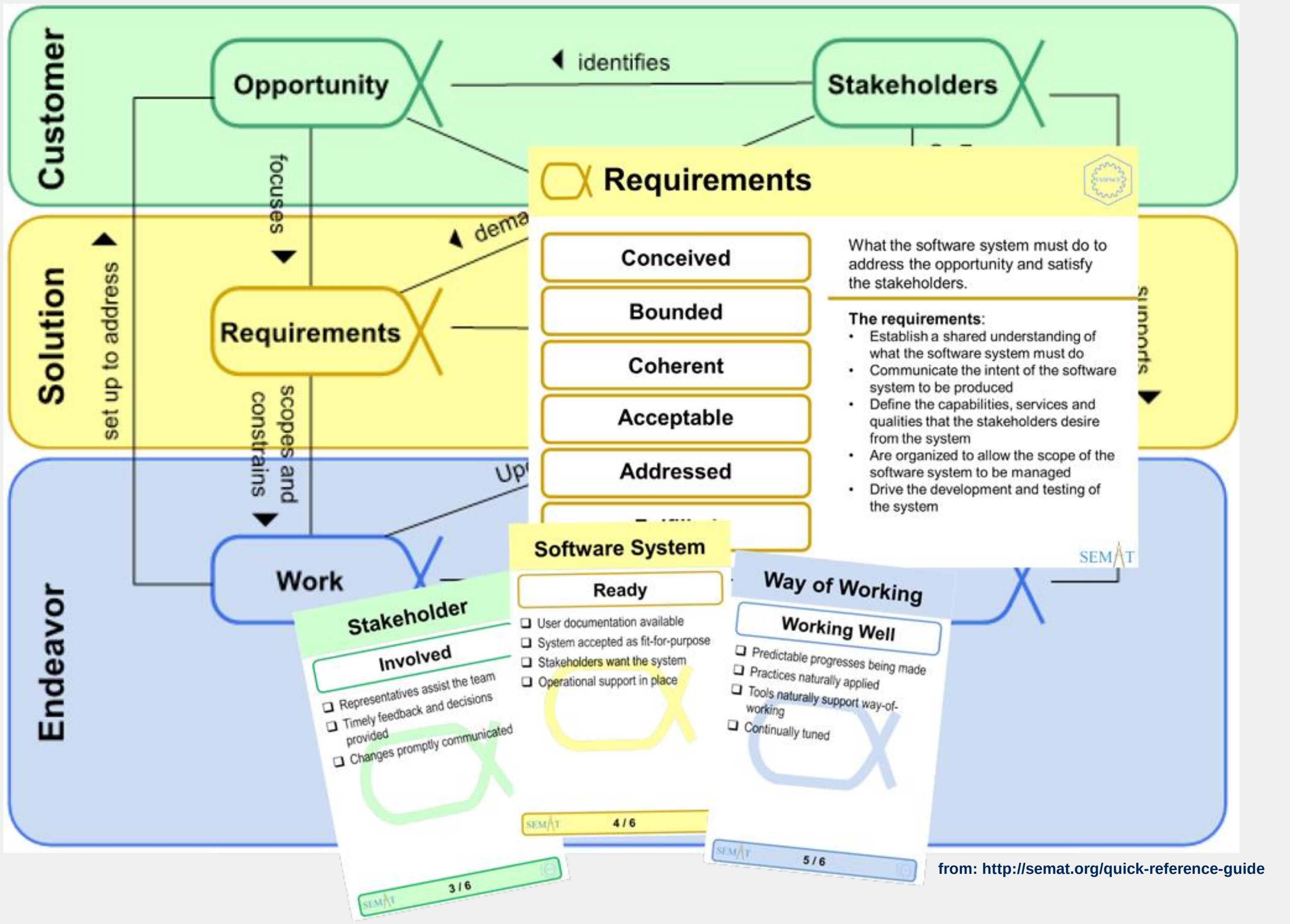
The Essence of Software Startups

- A practical tool for software startups
  - ”Survival kit” aimed especially at early-stage startups
  - Provides essential practices and tools for tracking progress
- Based on the Essence Theory of Software Engineering
- Based on scientific research but focus highly on the practicality of the artifact



Requirements

What the software system must do to address the opportunity and satisfy the stakeholders.

The requirements:

- Establish a shared understanding of what the software system must do
- Communicate the intent of the software system to be produced
- Define the capabilities, services and qualities that the stakeholders desire from the system
- Are organized to allow the scope of the software system to be managed
- Drive the development and testing of the system

Stakeholder – Involved

- ❑ Representatives assist the team
- ❑ Timely feedback and decisions provided
- ❑ Changes promptly communicated

Software System – Ready

- ❑ User documentation available
- ❑ System accepted as fit-for-purpose
- ❑ Stakeholders want the system
- ❑ Operational support in place

Way of Working – Working Well

- ❑ Predictable progresses being made
- ❑ Practices naturally applied
- ❑ Tools naturally support way-of-working
- ❑ Continually tuned

from: http://semat.org/quick-reference-guide

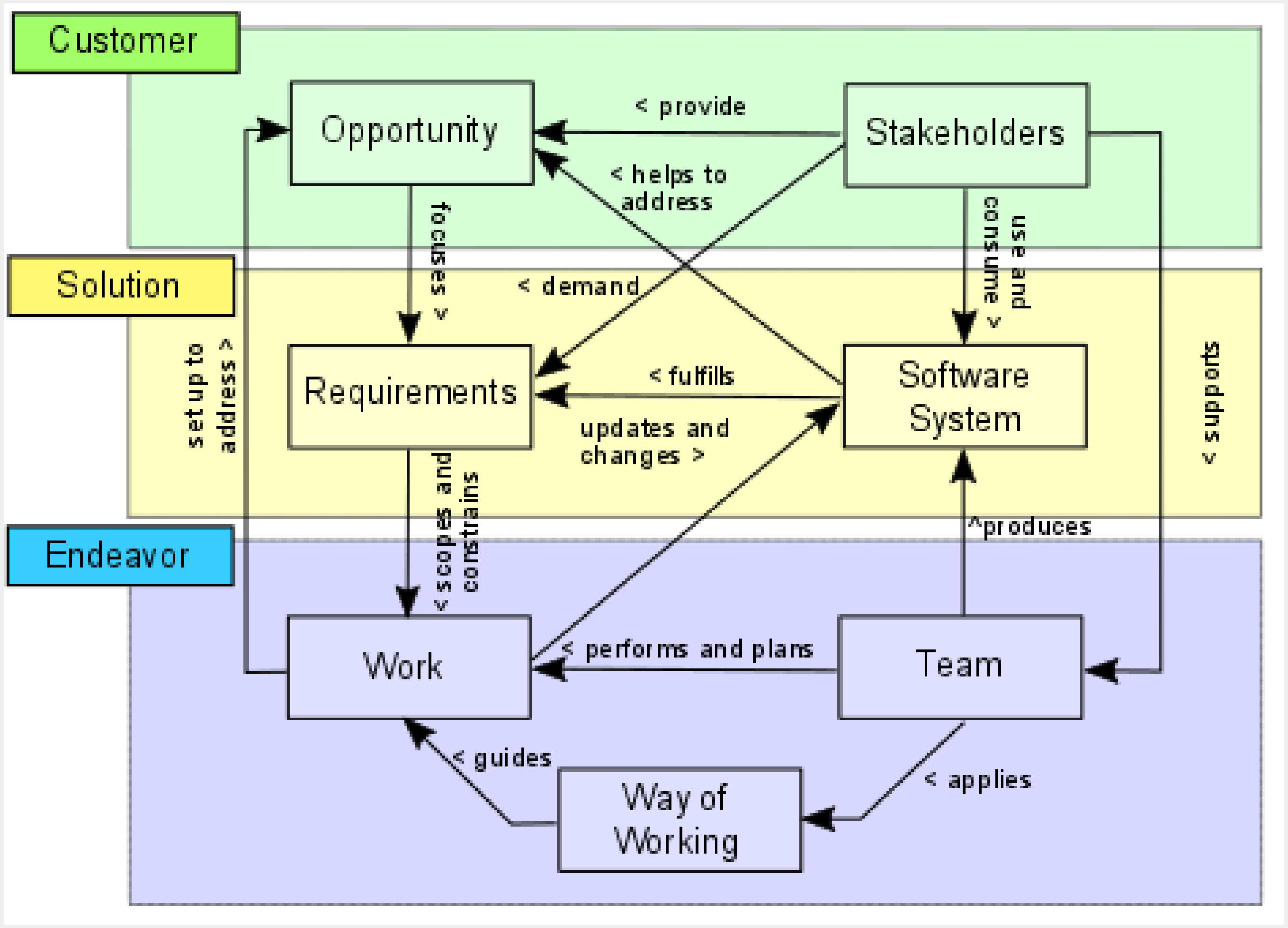
The Essence Theory of Software Engineering

- A method-agnostic software engineering tool for project use
  - / Progress management
  - / Method engineering
  - / Encourages communication and learning
- NOT a methodology or a model
  - / More akin to UML than e.g. SCRUM
    - → In fact SCRUM can be modeled using Essence

The Essence Theory of Software Engineering

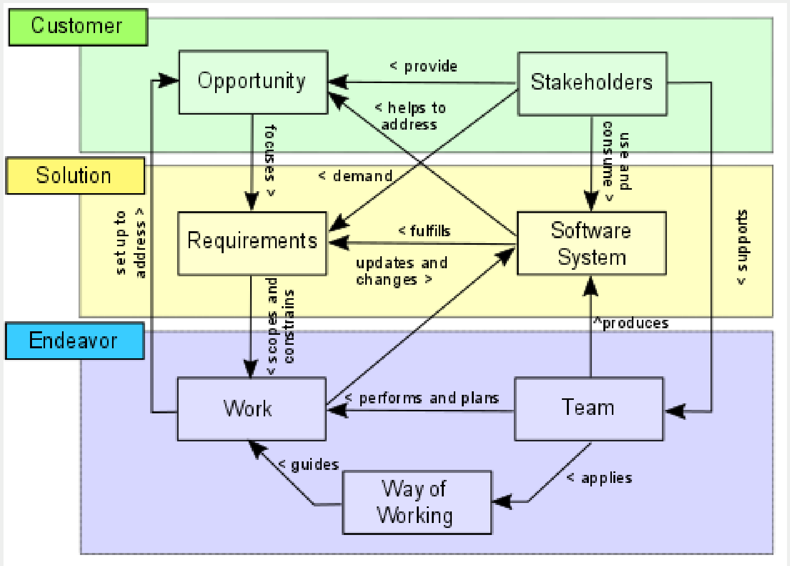
- Consists of
  - Kernel – contains elements present in every SE endeavor
  - Language – used to extend the kernel
- Kernel has three views
  - Alpha view
  - Competency view
  - Activity view
- Each view divided into three areas
  - Customer
  - Endeavor
  - Solution

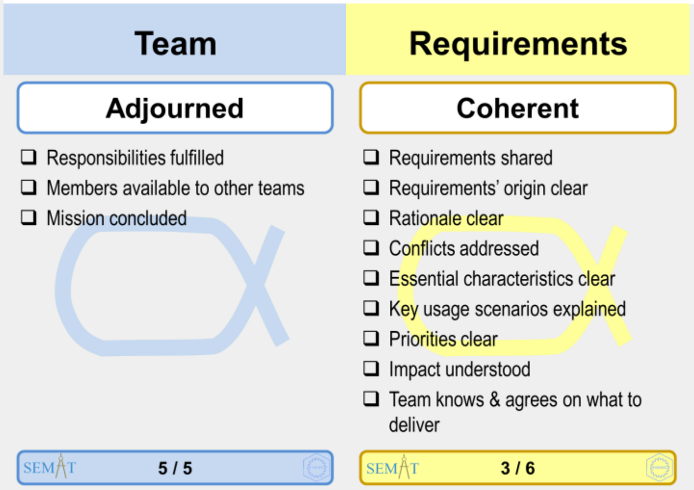
The Essence Kernel Alphas



The Essence Alphas

- Progress management
  - Project management (not comprehensive)
- Each alpha has states
  - Depicts progress from project start to finish





Team	Requirements
Adjourned	**Coherent**

Team – Adjourned
- ❑ Responsibilities fulfilled
- ❑ Members available to other teams
- ❑ Mission concluded

SEMAT 5 / 5

Requirements – Coherent
- ❑ Requirements shared
- ❑ Requirements' origin clear
- ❑ Rationale clear
- ❑ Conflicts addressed
- ❑ Essential characteristics clear
- ❑ Key usage scenarios explained
- ❑ Priorities clear
- ❑ Impact understood
- ❑ Team knows & agrees on what to deliver

SEMAT 3 / 6

from http://semat.org/alpha-state-cards-with-abbrev-checklists

The Essence Theory of Software Engineering

- Language is the defining feature
- A modeling tool that provides guidance for method engineering and carrying out a project
- Without extending the kernel, Essence is very general or ”bare bones”
- Complemented by a practice library from where one can pick practices to use

Problems with Essence

- Resource-intensive to adopt
- Lacks good learning resources
  - You have to read a book to get into it
- Suffers from current lack of practitioner interest
- Does not offer much practical advice, relies on the user to extend it
  - Too much ”Google this for more info” type content in e.g. practice cards

And what about those startups?

Startuplab JYU

LabStartup

Essence for inexperienced developers?

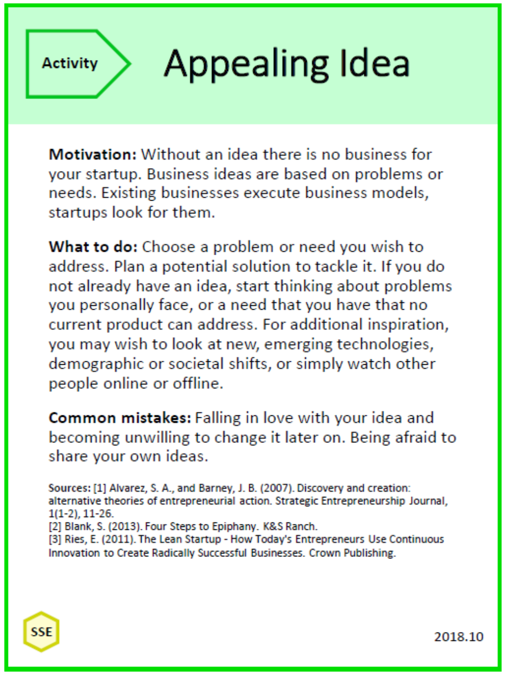
- We studied 100+ student teams using Essence in a project-based SE course
- These students found Essence useful because of their inexperience
  - Helped guide them into the right direction
  - Encouraged them to reflect on why they work the way they do
  - Provided ideas for how to work (practice library)
  - Helped them track progress on their project
- Startuppers are also often inexperienced -> maybe Essence could help them too?

The Essence for Software Startups?

- Startups engineer software differently
  - Ad hoc, various agile practices; highly varied
  - Technical debt
- Startups also need to work systematically
- Startups are very focused on learning
- Based on our experiences with students, early-stage (student) startups often feel lost on what to do at the **earlier stages**
- But…
  - Startups are not SE projects, they are firms
  - Essence is not very practical as is

The Essence of Software Startups

- Alphas are progress management tools -> closely related to metrics
  - We are studying software startup metrics
- Based on data so far:
  - New business alphas for the kernel
  - New area of concern for the kernel: business
- Focus on PRACTICE; activities and activity spaces
  - Currently focused on revamped cards
  - Similar to the practice library of Essence; ”startup essentials” (e.g. vs. their ”Agile essentials”)
  - Cards provide guidelines on what to do at the earlier stages
  - Currently tested in the Lean Startups course of 2018 at JYU
  - **Early-stage** software startups


Activity

Appealing Idea

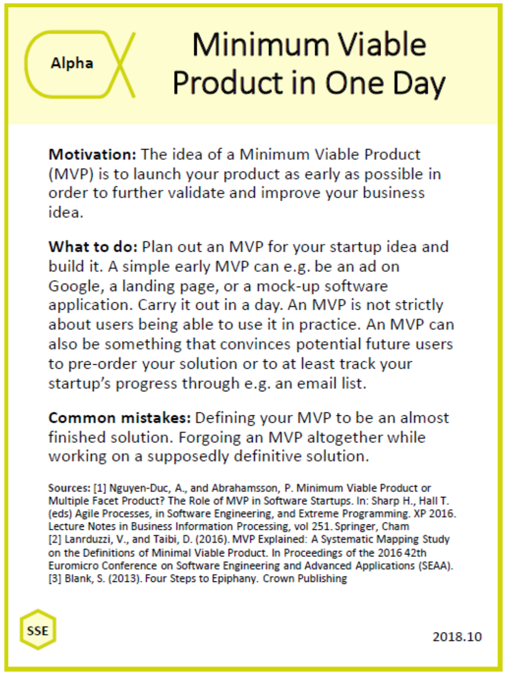
Motivation: Without an idea there is no business for your startup. Business ideas are based on problems or needs. Existing businesses execute business models, startups look for them.

What to do: Choose a problem or need you wish to address. Plan a potential solution to tackle it. If you do not already have an idea, start thinking about problems you personally face, or a need that you have that no current product can address. For additional inspiration, you may wish to look at new, emerging technologies, demographic or societal shifts, or simply watch other people online or offline.

Common mistakes: Falling in love with your idea and becoming unwilling to change it later on. Being afraid to share your own ideas.

Sources: [1] Alvarez, S. A., and Barney, J. B. (2007). Discovery and creation: alternative theories of entrepreneurial action. Strategic Entrepreneurship Journal, 1(1-2), 11-26.
[2] Blank, S. (2013). Four Steps to Epiphany. K&S Ranch.
[3] Ries, E. (2011). The Lean Startup - How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Publishing.

SSE 2018.10


Alpha

Minimum Viable Product in One Day

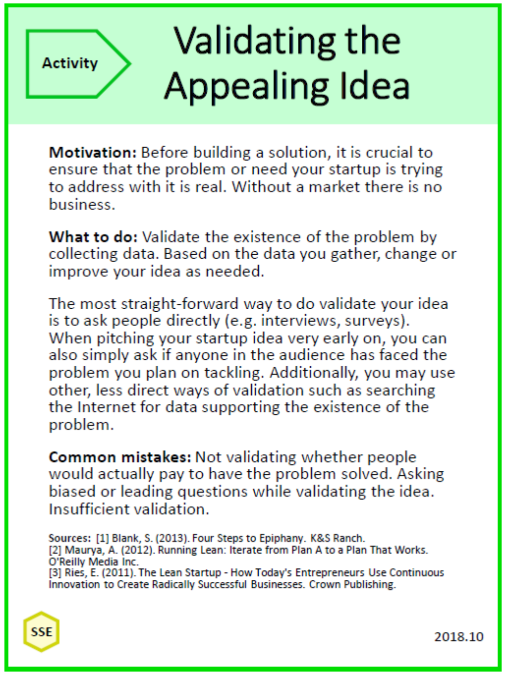
Motivation: The idea of a Minimum Viable Product (MVP) is to launch your product as early as possible in order to further validate and improve your business idea.

What to do: Plan out an MVP for your startup idea and build it. A simple early MVP can e.g. be an ad on Google, a landing page, or a mock-up software application. Carry it out in a day. An MVP is not strictly about users being able to use it in practice. An MVP can also be something that convinces potential future users to pre-order your solution or to at least track your startup's progress through e.g. an email list.

Common mistakes: Defining your MVP to be an almost finished solution. Forgoing an MVP altogether while working on a supposedly definitive solution.

Sources: [1] Nguyen-Duc, A., and Abrahamsson, P. Minimum Viable Product or Multiple Facet Product? The Role of MVP in Software Startups. In: Sharp H., Hall T. (eds) Agile Processes, in Software Engineering, and Extreme Programming. XP 2016. Lecture Notes in Business Information Processing, vol 251. Springer, Cham
[2] Lanrduzzi, V., and Taibi, D. (2016). MVP Explained: A Systematic Mapping Study on the Definitions of Minimal Viable Product. In Proceedings of the 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA).
[3] Blank, S. (2013). Four Steps to Epiphany. Crown Publishing

SSE 2018.10


Activity

Validating the Appealing Idea

Motivation: Before building a solution, it is crucial to ensure that the problem or need your startup is trying to address with it is real. Without a market there is no business.

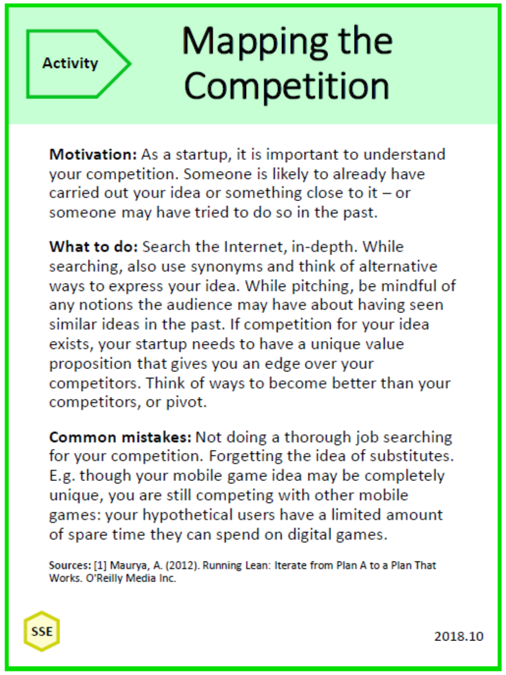
What to do: Validate the existence of the problem by collecting data. Based on the data you gather, change or improve your idea as needed.

The most straight-forward way to do validate your idea is to ask people directly (e.g. interviews, surveys). When pitching your startup idea very early on, you can also simply ask if anyone in the audience has faced the problem you plan on tackling. Additionally, you may use other, less direct ways of validation such as searching the Internet for data supporting the existence of the problem.

Common mistakes: Not validating whether people would actually pay to have the problem solved. Asking biased or leading questions while validating the idea. Insufficient validation.

Sources: [1] Blank, S. (2013). Four Steps to Epiphany. K&S Ranch.
[2] Maurya, A. (2012). Running Lean: Iterate from Plan A to a Plan That Works. O'Reilly Media Inc.
[3] Ries, E. (2011). The Lean Startup - How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Publishing.

SSE 2018.10


Activity

Mapping the Competition

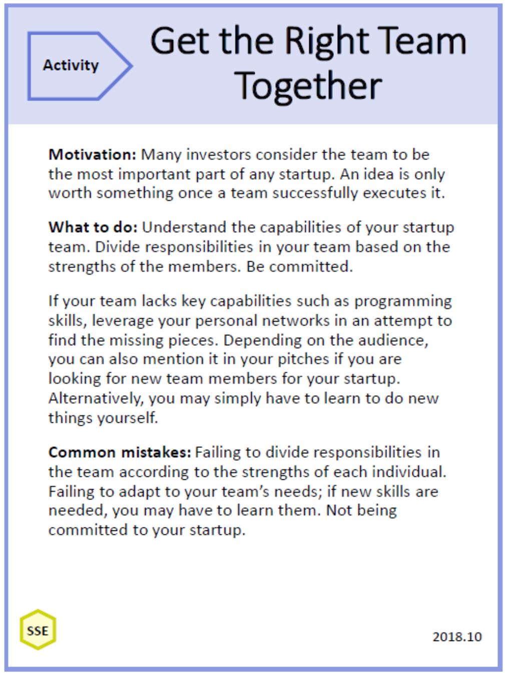
Motivation: As a startup, it is important to understand your competition. Someone is likely to already have carried out your idea or something close to it – or someone may have tried to do so in the past.

What to do: Search the Internet, in-depth. While searching, also use synonyms and think of alternative ways to express your idea. While pitching, be mindful of any notions the audience may have about having seen similar ideas in the past. If competition for your idea exists, your startup needs to have a unique value proposition that gives you an edge over your competitors. Think of ways to become better than your competitors, or pivot.

Common mistakes: Not doing a thorough job searching for your competition. Forgetting the idea of substitutes. E.g. though your mobile game idea may be completely unique, you are still competing with other mobile games: your hypothetical users have a limited amount of spare time they can spend on digital games.

Sources: [1] Maurya, A. (2012). Running Lean: Iterate from Plan A to a Plan That Works. O'Reilly Media Inc.

SSE 2018.10


Activity

Get the Right Team Together

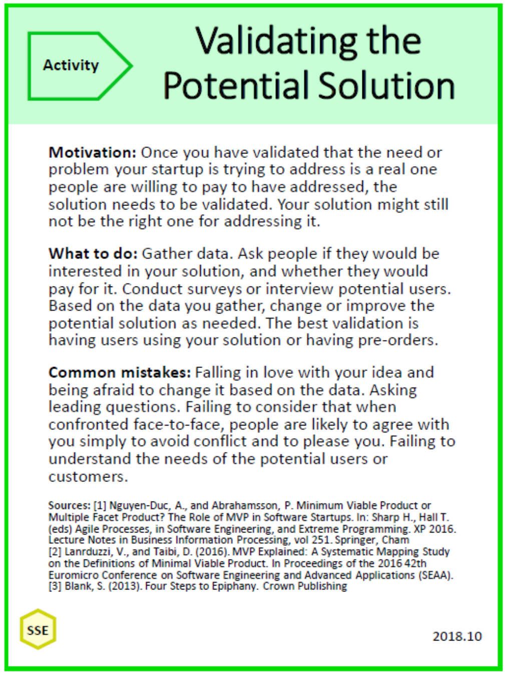
Motivation: Many investors consider the team to be the most important part of any startup. An idea is only worth something once a team successfully executes it.

What to do: Understand the capabilities of your startup team. Divide responsibilities in your team based on the strengths of the members. Be committed.

If your team lacks key capabilities such as programming skills, leverage your personal networks in an attempt to find the missing pieces. Depending on the audience, you can also mention it in your pitches if you are looking for new team members for your startup. Alternatively, you may simply have to learn to do new things yourself.

Common mistakes: Failing to divide responsibilities in the team according to the strengths of each individual. Failing to adapt to your team's needs; if new skills are needed, you may have to learn them. Not being committed to your startup.

SSE 2018.10


Activity

Validating the Potential Solution

Motivation: Once you have validated that the need or problem your startup is trying to address is a real one people are willing to pay to have addressed, the solution needs to be validated. Your solution might still not be the right one for addressing it.

What to do: Gather data. Ask people if they would be interested in your solution, and whether they would pay for it. Conduct surveys or interview potential users. Based on the data you gather, change or improve the potential solution as needed. The best validation is having users using your solution or having pre-orders.

Common mistakes: Falling in love with your idea and being afraid to change it based on the data. Asking leading questions. Failing to consider that when confronted face-to-face, people are likely to agree with you simply to avoid conflict and to please you. Failing to understand the needs of the potential users or customers.

Sources: [1] Nguyen-Duc, A., and Abrahamsson, P. Minimum Viable Product or Multiple Facet Product? The Role of MVP in Software Startups. In: Sharp H., Hall T. (eds) Agile Processes, in Software Engineering, and Extreme Programming. XP 2016. Lecture Notes in Business Information Processing, vol 251. Springer, Cham
[2] Lanrduzzi, V., and Taibi, D. (2016). MVP Explained: A Systematic Mapping Study on the Definitions of Minimal Viable Product. In Proceedings of the 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA).
[3] Blank, S. (2013). Four Steps to Epiphany. Crown Publishing

SSE 2018.10

The (Software) Startup Cards

- Based on existing research (mostly by the network), and practitioner literature
- Information mapping: every card has the same layout
  - Motivation
  - What to do
  - Common mistakes
  - (Sources)
- More Essence-inspired than Essence
  - No kernel or language so far
- Cards seem to be useful so far?

Data Collection

- How to evaluate this?
- Current plans
  - / Observation, mentoring; tracking progress and how they use the cards
    - → What metrics to actually focus on?
  - / Interviews / survey after course?
    - → Potentially the best and simplest way to measure if it was useful is to ask directly…

Current State?

- VERY early testing
- Cards are not the theory, merely a synthesis of existing research and practitioner literature
- The kernel and the language are left out for now; alphas are not a focus
  - Maybe later in the course?

The Endgame?

- Is this just another consulting card deck?
- How to make this actual theory **while also** keeping it practical?
- This is a PhD
  - / Most value for the academia comes from the papers comprising it
    - → Metrics papers etc.
  - / The artifact being designed here is iteratively created using the Design Science Research Methdology (DSRM) for Information Systems
    - →This is just the very beginning

References

- Ghanbari, H. (2017). Investigating the causal mechanisms underlying the customization of software development methods. Uni. of Jyväskylä: Jyväskylä Studies in Computing, 258.
- Jacobson, I., Ng, P., McMahon, P. E., Spence, I., and Lidman, S. (2012). The Essence of Software Engineering: The SEMAT Kernel. ACMQueue, 10, pp. 40-52.
- Kemell, K., Nguyen-Duc, A., Wang, X., Risku, J., and Abrahamsson, P. (2018). The Essence Theory of Software Engineering - Large-Scale Classroom Experiences from 450+ Software Engineering BSc Students. To be published in the proceedings of the 2018 International Conference on Product-Focused Software Process Improvement (PROFES2018).
- Kon, F., Cukier, D., Melo, C., Hazzan, O., and Yuklea, H. (2014). A Panorama of the Israeli Soft-ware Startup Ecosystem. SSRN: https://ssrn.com/abstract=2441157.
- Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T. and Abrahamsson, P. (2014). Software development in startup companies: A systematic mapping study. Information and Software Technology, 56, pp. 1200-1218.
- Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems, 24(3), 45-77.