



Aalto-yliopisto

NEPPI — Outsourcing

Pekka Nikander
Nov 23th, 2020

Learning goals

- Outsourcing: You should not try to do everything yourself
 - Defining core competencies and core value
 - Finding partners: quality or cost optimisation
 - Defining the rules for collaboration
 - Best practices for small company subcontracting
 - Support during product life cycle
- Approaches to collaborative resourcing
 - Open source software
 - Open source hardware

Outline

- Partnering
 - Competencies: Core and other
 - Partners: Hardware & software development
 - Example: Solu machines
 - Best practices
- Collaborative resourcing
 - Crowdsourcing
 - Open source software and hardware
 - Crowd funding

Outsourcing: Competencies

- Core competencies (Prahalad & Hamel, 1990):
 - Potential access to a wide variety of markets
 - Significant contribution to customer benefits
 - Difficult to imitate by competitors
- Everything else is non-core
 - But consider also Treacy and Wiersema's value disciplines
- Prahalad and Hamel (1990). "The core competence of the corporation." Harvard Business Review (v. 68, no. 3) pp. 79–91.
- Treacy, M., & Wiersema, F. (2007). The discipline of market leaders: Choose your customers, narrow your focus, dominate your market. Basic Books.

Competencies: Extreme example

- Concept design as the (only) core competence
- Everything else outsourced
 - User experience (UX) design
 - Software development
 - Hardware development
 - Packaging and shipping
 - Marketing and sales
 - User support
- Problem: How make it hard to imitate?

Core competencies: Senseg

- Product area: Electrostatic surface haptics
- Core competencies:
 - Haptics: physiology, psychology, ...
 - Space and energy efficient high voltage ($\sim 1\text{kV}$) generation
 - Thin film surface physics
- Problem: Too wide compared to resources

Outsourcing: Partnering

- Why to have a partner:
 - Easier to find and attract competence
 - Faster time to market
 - Needed less than a full-time person
- Why not to have a partner:
 - Harder and more expensive to manage
 - Partner has also other priorities
 - Paying with equity usually does not work

Partnering: Hardware development

- Very segmented market:
 - Mechanics design
 - Electronics design
 - Small scale PCB manufacturing
 - Small scale PCB assembly and testing
 - Large scale device production (ODM)
- Comprehensive hardware design subcontractors

Hardware development: 101 facts

- Hardware is still built in stages
 - Overall product design
 - Module-based prototype
 - 2 weeks – 2 months
 - 3–10 prototype PCB rounds
 - Each takes about a month (2 weeks minimum)
 - 1–3 pre-production prototype rounds
 - Each takes about a month
- Raising alternative: Build it around a Raspberry Pi

Hardware development: Examples

- Comprehensive hardware design
 - Haltian (Oulu): revenue ~6M, 70 employees
 - Wireless System Integration (Stockholm): similar?
- Electronics design
 - e-Hapines (Vantaa): ~50k, 1 part time employee
 - Convergens (Espoo): ~700k, 6 employees
- Small scale PCB assembly & testing
 - Jopaco (Lahti): ~3M, 9 employees
 - Sanmina (worldwide & Oulu): ~7B, 46000 employees

Hardware development: ODMs

- Original Design Manufacturer
 - Designs and manufactures a product
 - Device rebranded by another firm for sale
- Tier 1 (largest) ODMs:
 - Pegatron
 - Quanta Computer
 - Compal Electronics
- Some Tier 3 (smallish) ODMs:
 - Skyworth
 - Victory concept

Partnering: Software development

- Very segmented but differently
 - Horizontal rather than vertical
- Main areas
 - “Full stack” and parts of it
 - Mobile (iOS and Android)
 - Embedded
 - Artificial and augmented intelligence
- Lots of more specialised fields

Software development: 101 facts

- Personal productivity varies $> 100x$
 - Worst programmers: $< \text{less } 10 \text{ LoC/day}$
 - Best programmers: $> 1000 \text{ LoC/day}$
- Team productivity varies $> 10x$
 - Best scrum teams: $> 3000 \text{ LoC / day}$
 - Worst scrum teams: $< 100 \text{ LoC /day}$
- Agile methods today's de facto standard
 - You *can* change the specs on the fly

Software development: Examples

- Full stack / mobile houses:
 - Reaktor: 67M, 400 employees
 - Futurice: 62M, 530 employees
 - Codento: 4M, 37 employees
- Embedded:
 - Etteplan: 240M, 3000 employees
 - Offcode: 1M, 12 employees

Outsourcing: Example

- Solu machines
 - Hardware outsourcing from scratch
 - Software in-house (team or ~5 people)
- Starting point:
 - We want a Computer as a Service
- Successful ending point:
 - Working very early product prototypes

Example: First round

- Chose a Finnish embedded design house
 - I knew them already
 - They believed they can do it
 - They were willing to work partially on equity
 - They had some experience with Qualcomm
- Result: It didn't work out (2 partially working prototypes)
 - Hardware cost for this round: ~150k
 - Qualcomm licence would have cost ~1M\$
 - The design house knew only industrial, not consumer

Example: Second round

- Worked with WSI (Wireless System Integration)
 - Recommended by NVIDIA
 - They believed they can do it
 - Had some experience with consumer products
- Result: Real working prototypes (~10 of them)
 - Hardware cost for this round: ~600k
 - Own “tablet”: altogether around 400k
 - Special square display, with luck only ~200k

Outsourcing: Best practices

- Get a partner, not just a subcontractor
 - You are too small and high risk anyway
 - Build gradually a relationship
 - Become an opportunity for them
 - Partnering ensure product lifetime support
- Hire someone who knows the field
 - If you don't, you pay a premium and fail
 - Personal trust is much better than best written contracts
- Try to find someone local
 - Only if specialisation requires, go further

Outline

- Partnering
 - Competencies: Core and other
 - Partners: Hardware & software development
 - Example: Solu machines
 - Best practices
- Collaborative resourcing
 - Crowdsourcing
 - Open source software and hardware
 - Crowd funding

Crowdsourcing

- Use Internet to "outsource work to the crowd"
- Generic term with very wide variation
 - E.g. open source may be crowdsourcing
- May be peer production or not

- Most interesting: commons-based peer production

- Howe (June 2, 2006). "Crowdsourcing: A Definition". Crowdsourcing Blog.
- Benkler and Nissenbaum. "Commons-based peer production and virtue." Journal of political philosophy 14.4 (2006): 394-419.

Collaborative resourcing: Open source software

- In 1950s–60s most software was open source
- 70s–90s were prime time for closed source
- Today many new major projects are open source
 - Main exception: Pure cloud backend
- Reasons:
 - Accepted good licensing modes
 - Customer expectations
 - Community benefits
 - Relative cost of *entering* an OSS project has risen

Open source: Market reality

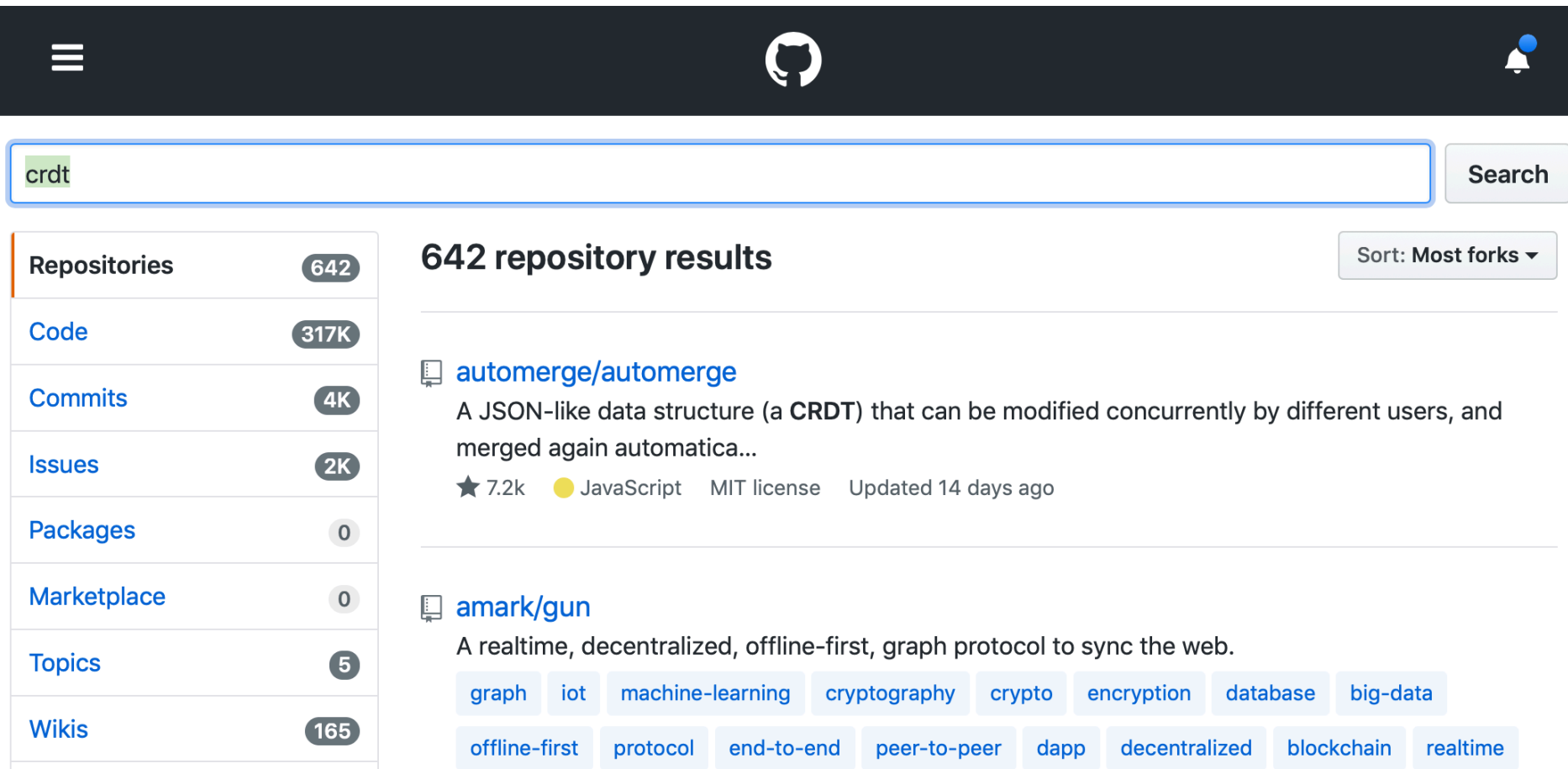
- For most common ICT problems, there is at least one Open Source solution
 - How to find it?
 - Is it good enough quality wise?
- If there isn't one,
 - Chance is that the research is still going on
 - E.g. collaborative editing 5 years ago
 - Or it is monetised as a backend platform

Open source: IPR and licenses

- Whoever writes the software, owns the copyright
 - E.g. with Linux, there are thousands
- Many projects have “Contributor License Agreement” requiring copyright transfer (but not e.g. Linux)
- **Open source license** gives others access
 - Different licenses have different terms
 - FSF Gnu Licenses (GPL, LGPL et al) require source code even from commercial binary-only vendors
 - Other licenses are usually more business friendly
 - Most registered with Open Source Initiative (OSI)

Open source: Finding software

- Search github
 - May need to iterate with keywords...
- Look at forks, stars, and activity
 - > 1000 stars, > 100 forks, still active



The screenshot shows the GitHub search interface. At the top, there is a search bar containing the text 'crdt' and a 'Search' button. Below the search bar, the left sidebar shows navigation options: Repositories (642), Code (317K), Commits (4K), Issues (2K), Packages (0), Marketplace (0), Topics (5), and Wikis (165). The main content area displays '642 repository results' with a 'Sort: Most forks' dropdown. The first result is 'automerge/automerge', described as a JSON-like data structure (a CRDT) that can be modified concurrently by different users, and merged again automatically. It has 7.2k stars, is written in JavaScript, uses the MIT license, and was updated 14 days ago. The second result is 'amark/gun', described as a realtime, decentralized, offline-first, graph protocol to sync the web. It has several tags: graph, iot, machine-learning, cryptography, crypto, encryption, database, big-data, offline-first, protocol, end-to-end, peer-to-peer, dapp, decentralized, blockchain, and realtime.

crdt

Search

Repositories 642

Code 317K

Commits 4K

Issues 2K

Packages 0


Marketplace 0

Topics 5


Wikis 165


642 repository results

Sort: Most forks ▾

 [automerge/automerge](#)

A JSON-like data structure (a **CRDT**) that can be modified concurrently by different users, and merged again automatica...

★ 7.2k  JavaScript MIT license Updated 14 days ago

 [amark/gun](#)

A realtime, decentralized, offline-first, graph protocol to sync the web.

graph iot machine-learning cryptography crypto encryption database big-data

offline-first protocol end-to-end peer-to-peer dapp decentralized blockchain realtime

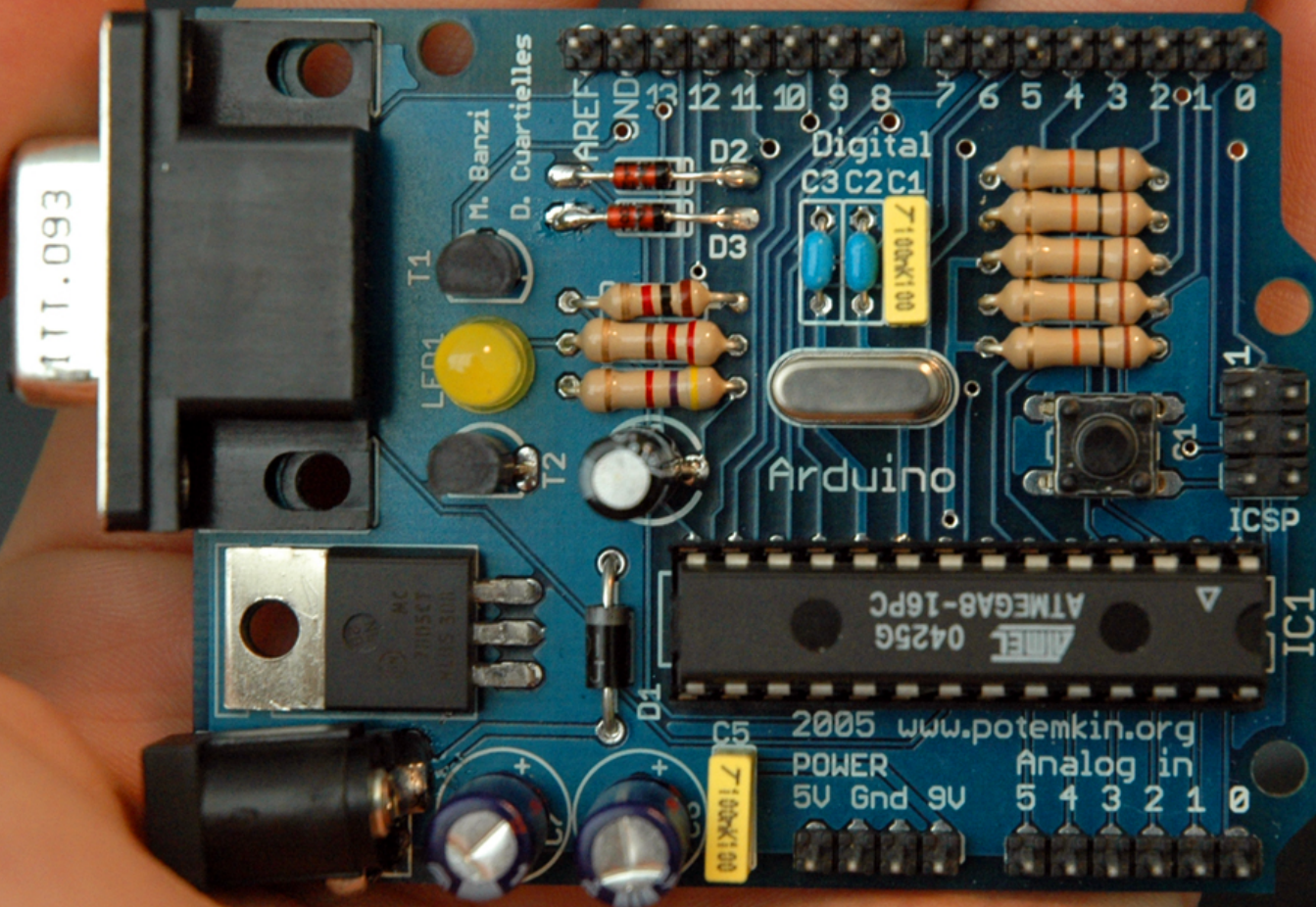
Open source: Adopting a solution

- Hire someone from the community
 - Must know the language and the field in large
 - May or may have worked with the SW itself
 - Preferably someone working actively
- If not possible, hire experienced professional
 - Expect 1– 6 months adoption time
 - Takes time to learn the software
 - Takes time to learn the community
- If not possible, forget in-house SW development

Collaborative resourcing: Open source hardware

- Relatively new phenomenon
 - Arduino from ~2006
 - More common since 2010
- More dispersed, less established than OSS
 - Many schematics and PCB layout designs exist
 - You still have to manufacture them yourselves!
- Changing the practices elsewhere in the industry
 - More chip vendors providing design examples as open source schematics and PCB design

Open source hardware: Arduino



Crowd funding

- Kickstarter
 - Best known platform
 - Some restrictions on supported countries
- IndieGoGo
 - Today larger than Kickstarter
 - Specialised in technology and hardware
- Note: Some consumers don't understand the difference between crowdfunding and purchases
 - Better to design one's campaign to be clear on this

Solu - A new breed of computing



Solu is the world's smallest computer with a cloud-linked OS, a revolutionary human-machine interface and a unique subscription model.

Follow along!

Created by

Solu Machines

609 backers pledged €219,543 to help bring this project to life.

 **Last updated** [September 21, 2017](#)

Summary

- Partnering
 - Competencies: Core and other
 - Partners: Hardware & software development
 - Example: Solu machines
 - Best practices
- Collaborative resourcing
 - Crowdsourcing
 - Open source software and hardware
 - Crowd funding