

Proposal for the Implementation of Engineering Processes

Denis Grelich

dgrelich@cstart.org

February 25, 2010

Version 1.3

<http://cstart.org/>



CSTART

Collaborative Space Travel and Research Team

Without thorough planning, the pyramids, the colosseum or the golden gate bridge would have never been completed. Feats like these are what makes humans unique on this planet. And space travel – the peak of civilization, *breaking the last frontier* – is impossible without efficient collaboration and effective problem solving. Part of this is a process called product management, in which goals and requirements are identified, solutions are developed and evaluated. Documentation fixes all progress and makes it available to others.

Contents

1	Definitions	2
2	Reasoning	2
3	Making sure you do not miss something and make the right decision	2
3.1	Processes	2
3.2	Requirements	3
3.3	Finding Solutions and Making Decisions	3
3.4	Documentation	4
4	References	4
5	Figures	5



Released into public domain.

1 Definitions

The following definitions will be used throughout this document; they are similar in engineering and software development, see [Leffingwell, 2001] and [Windley].

product, system the item that is going to be engineered

product management directing the engineering effort

project management allocation of resources, scheduling

process, method a defined set of 'rules' for doing things

The following are used when talking about the description of an engineering feat:

informal specification what the product is supposed to do in layman's terms: 'A rocket that flies into space and collects scientific data we can use ...'

vision colorful, optimistic description of the (future) product, a source of inspiration pointing out clear decision making criteria

mission the fundamental purpose, goal of the product, the desired level of performance – what is the system exactly supposed to do?

feature piece of textual description of a product's mission, rooted in the vision, mission and informal specifications, describing the system in an abstract way: 'reach the Kalman line', 'collect acceleration data', 'follow a planned trajectory'; there are easily dozens of those

critical information information without which problem analysis cannot be correctly performed and without which the development of a solution does not realistically make sense

requirements very specific characteristic, capability, quality or adherence to constraints, originating from features and other sources; the requirements list is strongly managed, changes are traceable but it's nonetheless an actively living, dynamic working document; there may easily be hundreds of them

design, solution there are many possible solutions for a problem that needs to be solved, i.e. for fulfilling a requirement; they have to be traceable to requirements, but there's no 1:1-mapping.

implementation actual implementation of a design or a solution, may be a drawing, a program or even a physical item

documentation concise collection of information on a decision in a form from which an onlooker can re-evaluate the decision and understand why certain aspects have been decided to be done in this particular way

2 Reasoning

Some people describe the way Open Source projects work using the "CADT" model – short for "Cascade of Attention-Deficit Teenagers." Zawinski [2003] It is characterized by doing things that are fun, or to 'scratch an itch.' Everyone does what he/she sees fit, and what should be done is specified in an informal way – mailing list or forum posts. It sort of works for software, in settings where there's no budget or allocation of resources problem.

Open Source software suffers from a set of peculiar problems though. A lack of planning and organization leads to 'spaghetti code,' feature bloat, bugs and overall poor design. Groupthink and bias are often a cause for bad design choices. What is seldom employed in the Open Source world are defined processes.

A successful undertaking needs careful, thoughtful planning and sticking to some rules on how decisions are made; it's all about communication and information management so that important details are not overlooked. All this requires a lot of effort and discipline, but is well worth it: many catastrophic failures are caused by bad working methods – the Challenger disaster was caused by a bad decision making process allowing for bias, groupthink and bad judgement. Feynman [1988]

3 Making sure you do not miss something and make the right decision

3.1 Processes

How should a (multidisciplinary) team approach a problem? Surely, an effective design process in engineering is an iterative one. It starts with spotting problems and deducing requirements. In the following step, alternative solutions are gathered and tested against the requirements so that either a preferred solution emerges or new problems are revealed, leading to another iterative step in requirement finding. To be able to work on complex problems, a structured approach is imperative, dividing problems into subproblems and solutions into subsolutions.

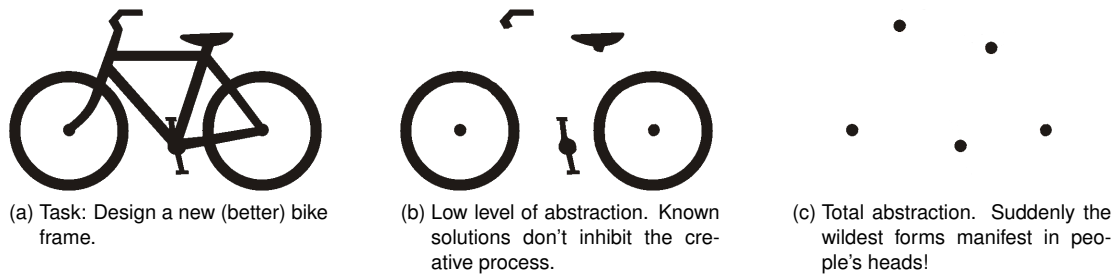


Figure 1: Abstraction in the design process.

Processes are easily enforced in a business environment with a clear hierarchical management system, but in a project of free people with free minds, it requires understanding and support of everyone involved! Also, people with the most different backgrounds regarding training and experience participate in the endeavour. This is why processes and good work habits must be communicated prominently and effectively.

3.2 Requirements

The initial step in any engineering process must be the definition of the problem and the determination of requirements a product must meet to be classified as 'success.' Only then can economical implication and the correct steps for a solution be assessed. Sutton/Thompson [Sutton and Thompson] write:

"By specifying information germane to the solution, the solver gains conceptual knowledge of the components of the problem. [...] The focus is on problem definition and objectives. Critical information needed to reach the objectives is identified. The identified critical information needs lead to data requirements. The economic scope and project objectives are revised as the available data and the data needs are assessed."

So, how does one efficiently find all relevant requirements without missing any critical details? The key is a structured, hierarchical approach. Figure 2 gives some ideas on methods to find requirements.

Requirement lists are not set in stone, the requirement list is a dynamic working document. Requirements change all the time when new insight is gained and new data is available, and if a requirement changes, it must be immediately clear what solutions are affected by this change in order to review them. Here, a requirement management system must be able to trace down all relationships between requirements, decisions and solutions.

3.3 Finding Solutions and Making Decisions

When it is believed that all or most critical requirements are known, all alternatives to the accomplishment of the objectives should be identified, and once identified, they should be evaluated to yield the most promising approaches to the problem at hand. This process could look like this:

1. Identify relevant requirements. The more critical requirements are found in an early development stage, the less likely is the need for a complete redesign later on.
2. Find possible solutions.
 - abstract away as far as possible (cf. fig. 1)
 - write down *everything*
 - use creative processes: catalogues, creativity methods, patent search
3. Decide on one solution as objectively as possible. This avoids scorn over disregarded ideas (don't underestimate the effect on team dynamics and motivation!) and the objectively best solution is chosen (cf. fig. 3). Justification for decisions must be solid: guesswork is not productive!
4. Implementation by team members.

Do not forget to document the entire process, all requirements that need to be met by the solution, all ideas (no matter how 'bad'), the decision making process and the aspects under which the solutions have been evaluated (maybe in tabular form), all calculations, all assumptions and *all sources*. This not only allows for self-evaluation, but this way the quality of a decision becomes apparent and the likelihood of bad design decisions is reduced. This documentation can be in the form of a short report and a wiki page and also in form of entries in the requirements/solutions/designs database with relationships to requirements. Just chat logs/forum entries are not enough, though.

Of course not all decisions can or should be team decisions. Implementation details are in the hands of the person who implements a design. Nonetheless, documentation is vital. Team members must be kept informed and up-to-date regarding the details and the reasoning behind them.

3.4 Documentation

Since meetings and presentations might be difficult due to the distributed nature of the team, briefings in form of forum entries, IRC room meetings or tele- or video conferences might be used to inform others of decisions, implementation details and the reasons behind them. Short reports of several pages length not only bring the team up-to-date but also create a wealth of educational material for newcomers. If team members don't feel comfortable navigating the work others have done already, participation will be reduced and reviewing this work is hard. This fact is especially important in multidisciplinary teams. Other aspects of good documentation include:

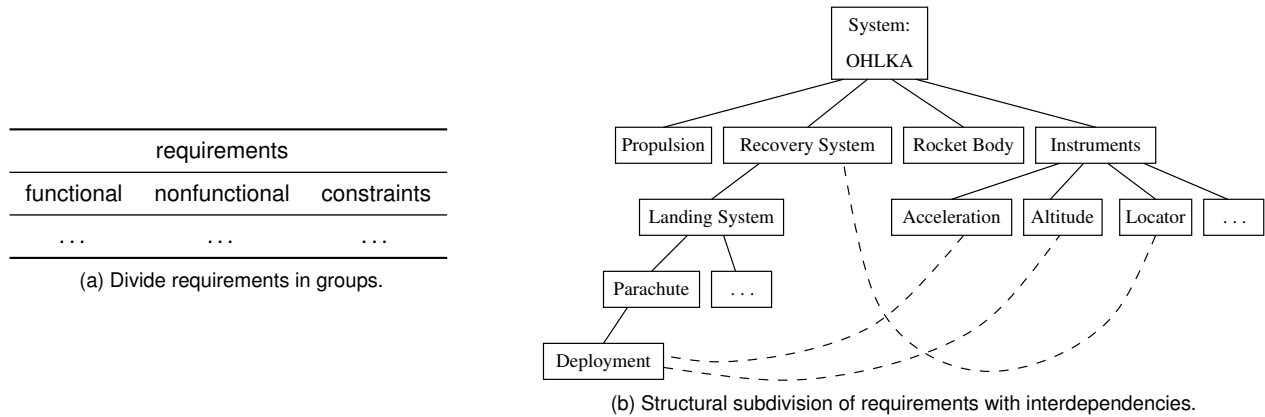
- No lost knowledge in case of ...
 - members leaving the team,
 - change in development direction.
- In case of reuse of old solutions, clearly stated justifications for decisions help immensely for adapting and modifying the solution; alternatives to the solution are also immediately visible.
- Provides rich material for education and lowered entry barriers for newcomers.
- A careful documentation process means thinking out thoughts completely which then helps in a thorough exploration of the problem space.
- Mistakes are found in the process of documenting and through review.
- Progress becomes visible and measurable.

Good documentation simply can't be stressed enough.

4 References

- Richard Feynman. *What Do You Care What Other People Think*. 1988.
- Dean Leffingwell. Features, use cases, requirements, oh my! *The Rational Edge e-zine*, 2001. URL http://www.therationaledge.com/content/dec_00/t_usecase.html.
- John Sutton and Robert Thompson. Multidisciplinary integration: A decision methodology and procedure for instruction. URL fie-conference.org/fie98/papers/1172.pdf.
- VDI Association of German Engineers. VDI 2221: systematic approach to the development and design of technical systems and products. 1993.
- Phillip J. Windley. The discipline of product management. URL <http://www.windley.com/docs/Product%20Management.pdf>.
- Jamie Zawinski. The CADT model, 2003. URL <http://www.jwz.org/doc/cadt.html>.

5 Figures



Properties/Constraints		technic-physical	human oriented	economical	normative	others
Lifecycle phase	Nr.	1	2	3	4	5
Development	1	production means	risk of injury	production and material costs	protective rights	trends
Distribution	2	transport, size	handling	storage costs	export regulations	transport means
Operation	3	function, maintenance	ergonomics	operating costs	directives, laws	exhibitions
Decommissioning	4	reuse	health hazards	recycling costs	environmental laws	politics

(c) Requirements search matrix loosely after [VDI Association of German Engineers, 1993].

Figure 2: Structured methods for acquiring requirements.

	specific thrust		price		handling		...	Sum	Rank
weighting	25%		25%		13%		...		
paraffin wax/N ₂ O	4	1.00	3	0.75	4	0.52		(2.27)	2
methane/LOX	2	0.50	4	1.00	2	0.26		(1.76)	3
polyethene/N ₂ O	4	1.00	4	1.00	3	0.39		(2.39)	1
...									...

score: 4 – good, 3 – moderate, 2 – bad, 1 – unacceptable

Figure 3: Possible form of a decision table for determining solutions worth developing. The sum of weighted scores determines the rank of a solution.