# An elastic organizational structure for software companies

* [Version](#scroll-bookmark-2)
* [Overview](#scroll-bookmark-3)
* [Motives](#scroll-bookmark-4)
  + [Situation](#scroll-bookmark-5)
  + [Conway's Law](#scroll-bookmark-6)
* [Objectives](#scroll-bookmark-7)
* [Structure](#scroll-bookmark-8)
  + [Groups](#scroll-bookmark-9)
  + [Work Flow](#scroll-bookmark-10)
  + [Hierarchical layer](#scroll-bookmark-11)
  + [Process](#scroll-bookmark-12)
  + [Use cases](#scroll-bookmark-13)
* [Products](#scroll-bookmark-14)
* [References](#scroll-bookmark-15)

# Version

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Author | Date | Changes |
| 0.1 | Jin Feng | 2019-7-14 | Initial Draft |

# Overview

We are facing lots of challenges from products development. Per Convey's law the organization's communication structure dictate the software design, so here we are trying to tackle challenges with an elastic organizational structure. In this article I want to propose an elastic organizational structure for software companies to adapt varieties of products and to maximize the resources uses.

In motives chapter I will give you situations we are facing and the reasons why we need an more effective structure.

In objectives chapter I will give you the properties that the designed organizational structure shall be have.

In structure chapter I will unveil the imagined effective structure and how it looks like. And also give you some typical scenarios to demonstrate how this structure woks for kinds of projects

In products chapter I will give you a new thinking to treat products thus the lists of products can be extended greatly.

in references chapter I will list all the documents cited by this article

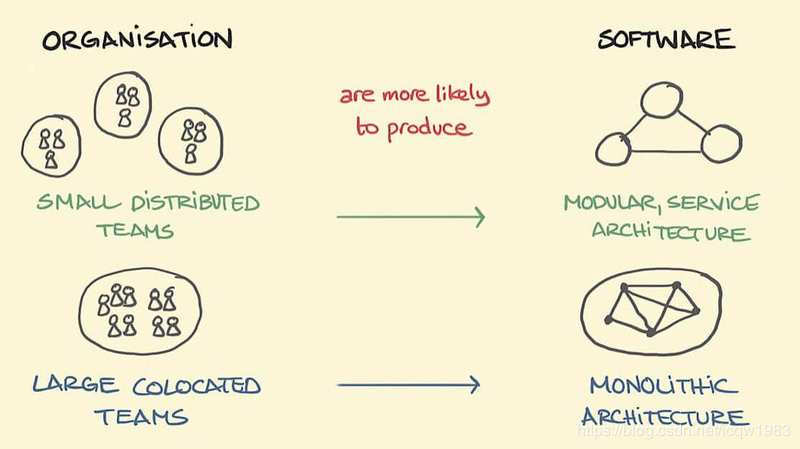
Before going on, I would like to give you the definitions of project and product used in this article. A product is a software or a hardware with software. A project is an managed activity to develop a product.

# Motives

## Situation

1. The requirements/design/implementation could be change
2. Lots of and kinds of projects shall be covered simultaneously. These projects could be mass productions, demos, POC, pre-development, internal, external, only applications, whole platform, e.t.c
3. Varieties of products emerge and disappear rapidly
4. Some products/projects cover hardware, bsp, framework, application and backend. Some might be only software components.
5. Our products are only IVI and V2X. It is too narrow
6. It is hard to recruit qualified software engineers on demand unlike the production workers, but the employees' utilization rate has peak and trough.
7. Some employees some projects are busy, the others might be leisure.
8. The skills and knowledge is hardly share among different of  products (such as TB0X/V2X and IVI). Actually they have lots of commons from software engineering's perspective.
9. We are lack of all-rounders and specialist.  Developers are valuables assets for our company. Since it is hard to employ many technical aces to be worth a ten. Division of labor on the basis of specialization is better choice.  Assigning specific functions to a junior one, and let he/she be an expert in this specialized areas. Fusing these ordinary engineers who is only expert in some modules can let system department be an unified expert from the view of external side.
10. High staff turnover
11. The software components are re-developed again and again in different projects of same product
12. The software components are re-developed again and again for different products
13. Too many communications between internals and externals
14. Puss the buck
15. Response is not quick
16. The work scope is not clear. If let it be, some modules will be dragged, communication cost will increase, dispute will occur, chaos will follow, and project will be delayed. Europe has ”The Empire was killed by iron nails”, and we have "A small leak will sink a great ship". So it is really necessary and critical to have a clear and definitive technical work scope and module assignments.

## Conway's Law



Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure[1][2].

Brooks recognized that the law has important corollaries in management theory. Here is one stated in the paper. Because the design that occurs first is almost never the best possible, the prevailing system concept may need to change. Therefore, flexibility of organization is important to effective design[1].

# Objectives

Here we list the properties the desired structures should have.

1. Flexible for design changes[1] and varieties of products
2. Scalable for different company sizes
3. Can cover varieties of projects/products occurred in parallel
4. Decouple the organizational structure from products and business
5. Every component or a combination of several components can be product and profitable
6. The resources especial human resources shall be shared
7. The skills and knowledge shall be shared among kinds of products
8. The division of work shall be clear-cut, each one being charged with specific responsibilities.
9. The talents pool shall be big enough and like a spindle to resist the high turnover
10. The company's growth is driven by the ability of R&D rather than marketing/sales
11. The common software components shall be abstracted, reused and evolved for kinds of projects and products.
12. A project manager is necessary whatever the project it is

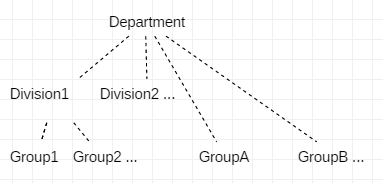
These attributes will be against the problems listed in motives chapter.

# Structure

As we know the organizational structure is an human group actually.  Some rules from sociology can guide our design:

* The R&D hierarchical layers should be no more than 3[11] ← SNAFU
* The direct subordinates should be no more than 9 members[9] ← Dunbar's Number
* Divide and conquer

With the limitations of above rules, here I show the general hierarchical layers and their descriptions. In theory the largest size we can support in one department is 10x10x10=1000 persons.



We call a collection of groups "division", and a collection of division is department. The purpose of  division and department is easy management.

|  |  |  |  |
| --- | --- | --- | --- |
| Layers | Description | Members | Comments |
| Group | A combat unit which is focus on one or several similar  tight coupled functions. | GL: Management, planning, interface, a little coding  TL: Internal architecture, code review, core coding, bug fix, coc  Developers: coding, bug fix, testing | GL and TL can be one person  Developers shall less than 10 persons |
| Division | Combine several groups for easy management | Manager: Management, coordination,  GLs: | TLs shall less than 10 persons |
| Department | Combine several Divisions for easy management | Director: Management, coordination, direction  Managers: | Managers shall be less than 10 persons |

The core of this structure is group. The purpose of division and department is for easy management.  So I will only describe group in following subsection.

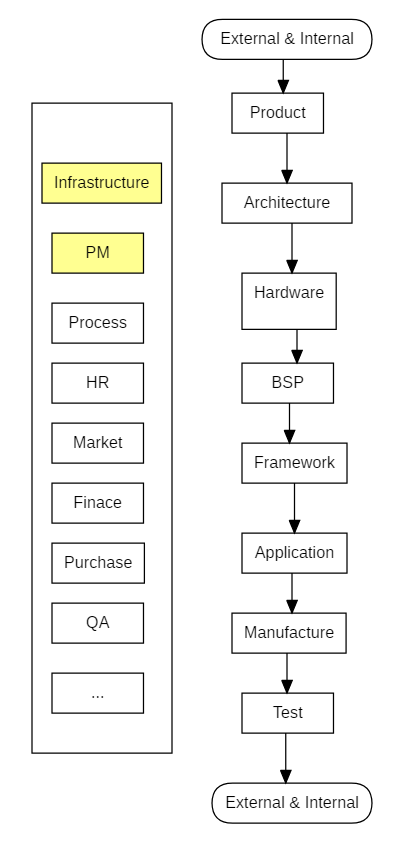
## Groups

The essence of this organizational structure is modular which is represented with form of group. A group:

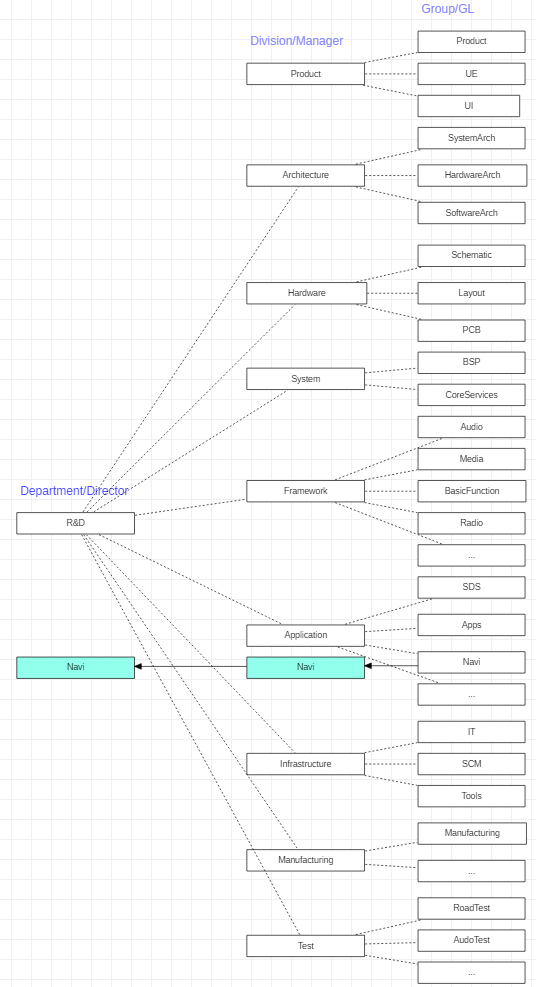
* is an aggregate of individuals who are cohesive works
* can be a functional  (such as BT group), or technical unit (BSP group), or management unit (such as PM group).
* is a combat team
* is a profit unit
* is spindle shape to resist turnover
* is a place to deposit  knowledge and skills of both technique and management.
* is an area to evolve reusable components for kinds of products and to implement COC (center of competence)

To setup so highly efficient group you can refer to the chapter 3 "A Surgical Team" of the book called "The Mythical Man-Month".  The surgeon shall be backed up for leaving.

## Work Flow



## Hierarchical layer



## Process

We take every outputs of activities such as POC, demo, pre-development as product whatever the maturity degree.

Each product shall be managed by project manager.

A product is produced via assembling the outputs of groups.

A project is the collaboration among groups coordinated by project manager.

## Use cases

IVI: todo

V2X: todo

OTA: todo

# Products

In theory we can take the output of a group or combination of outputs from several groups as a product. Thus theoratically the number of products is 2^n ( 2^n =C0n+C1n+...+Cnn)

Based on this thinking,  we can get the conclusions:

* The products list can extend  intensively for our sales person.
* The employees' utilization rate trough can be compensated by massive products

Now we can pick some promising products from outputs of groups

|  |  |  |
| --- | --- | --- |
| Product | Outputs of Groups | Description |
| IVI | All groups | Whole IVI product |
| V2X | hardware, bsp, networking, persistence... | Whole V2x product |
| Security solution | Architecture, networking | service and constant |
| BT | BT | software |
| Carplay/Carlife/Android Auto | SAL | software |
| OTA | OTA | The OTA client, protocol, and cloud setup and maintain |
| BSP development | BSP | outsourcing BSP development |
| APP development | Application | outsourcing application development |
| Auto testing bench | hardware, testing | mechanical+hardware+software |
| Automotive Ethernet | BSP, framework |  |
| ... |  |  |

# References

1. <http://www.melconway.com/Home/Committees_Paper.html>
2. <http://www.melconway.com/Home/pdf/committees.pdf>
3. <https://www.cnblogs.com/ghj1976/p/5703462.html>
4. <https://en.wikipedia.org/wiki/Conway%27s_law>
5. <https://blog.csdn.net/icqw1983/article/details/83782014>
6. <https://medium.com/@emma_j_saunders/why-organization-design-shouldn-t-be-nice-8f34d0a54751>
7. <https://www.heflo.com/blog/business-management/small-business-organizational-structure-examples/>
8. <https://smallbusiness.chron.com/flat-vs-hierarchical-organizational-structure-724.html>
9. <https://blog.csdn.net/samxx8/article/details/51930577>
10. <https://www.infoq.cn/article/K9qVBKjfzh3lFzzJ_MvF>
11. <https://rationalwiki.org/wiki/Discordianism#SNAFU_Principle>
12. [https://www.infoq.cn/article/every-architect-should-study-conway-law](https://www.infoq.cn/article/every-architect-should-study-conway-law/)
13. The Mythical Man-Month