Nikolay Ashanin
Lead Software Engineer at EPAM Systems. Interested in software architecture and team management.
Oct 1 · 8 min read

The Path to Becoming a Software Architect

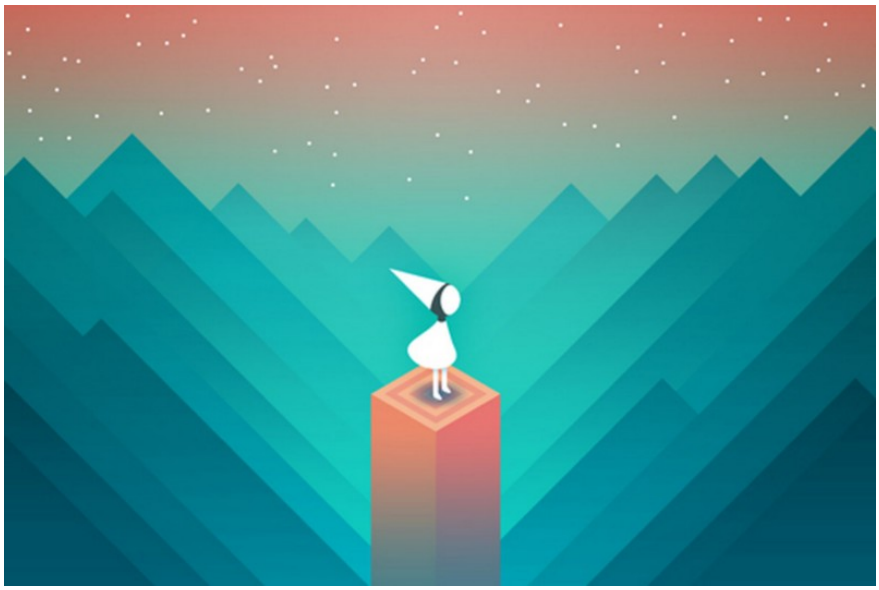

Image source: monumentvalleygame

Have you ever wondered what career opportunities a developer has? What directions are open, beyond what horizons to grow. And most importantly, where are developers beyond the age of 45? Is there a developer among your friends who is over 45? I personally know several developers beyond this age and many of them are hardcore programmers who even saw punch cards back in the day.

There are several career paths a developer might take:

● The first and obvious one is to **grow in the area in which you are working**. If you are a junior developer, then just grow to middle, then senior and lead roles.

● **Transition to another technology stack**. Actually, a big number of developers moved into the mobile area when iOS and Android OS gained ground.

● **Grow into a manager role**. As a developer, the greatest staffing issue I saw was the shortage of competent managers. Clever managers are expensive, hence they are scarce. If the manager has a technical

background, that will allow him to be on the same wavelength with the developers.

• **Become a software architect**. This direction will be considered in this series of articles.

• **Get out of IT**. Sometimes this happens. It is never too late to do what you like to do.

Yes, This Was My Path

In the past 8 years, I have worked with Java EE, then moved to iOS, and became a team lead. I managed various developer teams, including Android, and Web stacks. Created the architecture of the network layer for several services developed by the company, with sockets and REST API. I became acquainted with the managerial role and the prospect of growing in this direction while in the position of team lead for over two years. In my next role, my goal is to grow as a software architect.

For most developers, the function of the architect on the project is often unclear, so in this series of articles, I will try to find the answers to these related questions. Who is an architect, what is the scope of responsibilities, and how to grow in this direction and outline and action plan for myself and beginners wanting to move along this path.

Who Can Benefit from This?

This series of articles will help you if you belong to one of the following categories:

• **IT developer or engineer**. You are still growing as a developer, but you are looking ahead and planning your career. Even if the goals are initially vague, a person who consciously sets strategic goals will reach them much quicker than a person who does not plan where she is heading.

• **Team leader, lead software engineer**. You are at the highest stage of the software development discipline. To grow further, you have a choice to either learn one more stack of technologies, pursue a career outside software engineering, or to become a software architect.

• **Software architect**. You recently took this position, or have been working in this field for a long time. Perhaps one of the main qualities of such a specialist is the understanding that there are always areas

that a person does not know and that the learning process is continuous.

• **IT manager**. Although you are a manager, you understand perfectly well that you should at least approximately understand what your subordinates or colleagues are doing. The acute problem of management is the technical incompetence of the manager in the field in which he or she is managing.

Who is an Architect?

Before moving on to more specific questions, it is necessary to define the software architect role and it responsibilities.

> *A **software architect** is a software expert who makes high-level design choices and dictates technical standards, including software coding standards, tools, and platforms. The leading expert is referred to as the chief architect. (Wikipedia, The Free Encyclopedia, s.v. "Software architect", https://en.wikipedia.org/wiki/Software_architect)*

Like most high-level positions, there are no clear criteria that define this role. However, it is possible to define a number of responsibilities and qualities that contribute to the career of the architect.

First, let's consider the **characteristics of the architect**:

• **Communicability**. Having talked with many software architects, I heard that it is one of the most important characteristic of this specialist. During the working day, they have to talk with customers in the language of business, managers of all levels, business analysts and developers. If you have a natural charisma and you know how to convince people, then this will be a huge plus, as it is very important to explain your actions correctly. Architects are laconic, eloquent and competent speakers. The software architects with whom I spoke have highly developed skills in communication and persuasion. Another reason why this characteristic is most important is that the architect in this role participates in most discussion making processes, and often compromises must be reach that are acceptable and beneficial for all involved parties.

• **Broad and deep technical knowledge**. This should be obvious since one cannot become a software architect with a medical background. In addition, the architect usually has knowledge in several technological stacks at a decent level, and should have a good understanding of a few

other ones. The software architect should also be prepared to compose a large number of technical documentation, reports and diagrams.

• **Responsibility**. You should understand that architect decisions are usually the most expensive. Therefore, a person in this position should take the most responsible approach to his work and to the decisions made. If the developer's error costs a couple days of work of one person, then the architect's mistake can cost person-years on complex projects!

• **Stress resistance**. You will have to make decisions because in this role, you will be asked to do so and you will need response. You will be working with different people from different areas, and you will have to deal with rapidly changing demands or even with changing business environments. Therefore, it is necessary to be ready for stress and to look for some ways to escape negative emotions. Work is always more pleasant when it brings pleasure. So if you choose this role only for the money, then think again.

• **Management skills**. This includes both organizational and leadership skills. The ability to lead a team, which may be distributed and composed of very different specialists, is essential.

• **Analytic skills**. Even if a specialist has a wide erudition in technology, he has tried many things on his own or participated in projects of various types, this does not guarantee that he can easily change the style of thinking to architect. One of the most important tasks is the ability to represent an abstract problem in the form of some finite real object of the system, which developers are already evaluating, designing and developing. Great communications skills are essential to clearly represent the abstraction in the form of the final system to the members of the team and the customer. It will be necessary to clearly communicate to both business and development, what is still to be done.

If we talk about the responsibilities of the architect, then here is the perfect example from 19th century about bridge construction. At that time, the tests of the newly constructed bridge were the following: the key group of engineers, architects and workers stood under the bridge while the first vehicles were on it. Thus, they staked their lives upon the construction and the strength of the structure. So if there is a question —what is the responsibility of the software architect on the project? The answer is, he is responsible for everything.

If you give up loud and beautiful phrases, then the architect's work includes:

• Identifying the stakeholders on the project.

• Identifying business requirements and requirements of the stakeholders on the project.

• Designing the entire system based on the received requirements.

• Choosing the system architecture and each individual component of this system at a high level.

• Choosing the technologies for the implementation of each component and connections between the components.

• Architectural review. Yes, yes, it exists.

• Code-review.

• Writing project documentation and its support.

• Creating unified development standards in the company.

• Controlling the architecture during the next iteration of the system release.

This is only a subset of the software architect's responsibilities. The most important responsibility is complete technical support of the project from the moment of inception, through product release, to development of enhancements. And supporting the next releases. It will be necessary to switch a lot between different tasks during the working day.

How to Become a Software Architect?

To begin with, it is important to define milestone goals that lead to achieving your strategic goal of becoming a software architect. For me, such **goals for the next six months** are:

• **Understand and try several technological stacks**. My current knowledge is concentrated in the field of iOS. It is necessary to try Android, several server languages, to start python, and refresh Java EE skills. The architect is a full-stack developer, so it is important to have a broad technical knowledge.

• **Reading literature**. It is important to determine the most valuable books and articles that will help to grow in this direction. Usually the most effective way to find such literature is ask other professionals in this field for their recommendation. In one of the future articles, I plan to give you such a list of literature.

• **Find a mentor**. It is desirable to find a software architect at your current place of employment. It is always easier to get experience from a trained specialist than to start considering a certain area from scratch. It is important to be prepared to ask good questions from your mentor.

• **Study courses/obtain certificates**. There are many courses and certificates available, but only a few are worth their money, and the higher level courses cost a lot of money. Personally, I have attended the architectural courses of Luxoft (http://www.luxoft-training.com/it-course/ARC-001/), which have proven to be a worthy investment. It is extremely important that the lecturer of the course be a professional in this field and be able to answer questions. As for certificates, before starting, it is best to understand whether there are authoritative certification systems for architects and whether it is worthwhile obtaining the certification. This point I will discuss in a future article of this series.

One of the most important parts is a clear and stable plan review. What has been done, what should be reviewed, and where to accelerate or which goal to remove as useless.

Check Your Readiness Level

If you are interested in this introductory article from the series on how to become a software architect, or if you suddenly have thoughts to try this path, it is worth making sure that you really want it.

Firstly, people are afraid of everything new. A new position, new kind of stress, as opposed to the comfortable status quo. Of course, the choice is not always unambiguous and depends on how much you are willing to change something in your life. At the same time, it can depend not only on you, but also on the family, your financial commitments, parents and other factors.

Secondly, this path takes several years. The process of becoming a software architect does not happen overnight. As a team lead, I realized what to do and how to deal with stress only a year after I was appointed

to an official position. At the same time six months before that, I performed it unofficially. One software architect I know said that he understood what his responsibilities are 18 months after he was promoted to this role. Such intervals of time are normal and you need to understand whether you are ready to move in this direction. Even if you do not have a stable plan ready, it is better to start taking small steps that move you ahead, rather than remaining in the same place.

Standing in the same place in IT is a synonym for stagnation and personal fetters in life.