Setting up an architecture department [closed]

Asked 16 years, 3 months ago Modified 12 years, 10 months ago

Viewed 5k times

13

Closed. This question is opinion-based. It is not currently accepting answers.

Want to improve this question? Update the question so it can be answered with facts and citations by editing this post.

Closed 5 years ago.

Improve this question

Some context upfront:

Imagine a 200+ developers company finally setting up a more or less independent architecture team/department. The software portfolio consisting of 20+ "projects"/applications of varying sizes in production was taken care of by team-leads/technical-leads, who were responsible for and in charge of the projects "architecture" as well.

Out of the necessity to consolidate and control the architecture and enable certain needed large reworks on

the systems as a whole, in addition of the all so needed knowledge exchange, the company decided to set-up an architecture department.

- What are the **DO**s and **DON'T**s of such an undertaking?
- Who are the people making up such an architecture team?
- What should be their responsibilities?
- What's out of their scope?
- What are the useful transition strategies for the company?
- How to prevent those wry looks every time someone even mentions "the architecture team"?
- Did your company undergo such a change already successfully?
  Why did it fail?
  Why was it successful?

That's should **not** be a discussion on "What is architecutre?"(which is very closely related ;).

The really interesting points would be acceptable/realistic maybe even frictionless ways to install such a team, besides of course some warnings regarding battles better not to be even started.

architecture

Share

Improve this question

Follow

edited Jan 31, 2012 at 14:56

skaffman

403k ● 96 ● 824 ● 774

asked Sep 23, 2008 at 18:42

pointernil

608 ● 8 ● 17

As I illustrates in my answer below, there is no such thing as *one* architecture team for an organization of this size. Too much topics too cover. – VonC Sep 24, 2008 at 5:44

2 Almost 5 years later, I'm curious to know what route was chosen, and how it turned out to be :) – Sergey Apr 5, 2013 at 3:59

8 Answers

Sorted by: Highest score (default)

7

Here are a few issues that should be thought about:

- What is the exact mandate for the architecture team?
- What is the architecture team's deliverable? A framework, guidelines, implementation help... Or are they just Architecture Astronauts?
- Is this only for applications going forward, or will this be a backport?
- Who will be responsible for backporting? (And we mean budget here...)

- Will there be resources allocated to testing the backports?
- Does the Architecture Team have real muscle, or will management's will fold when the first group grouses about the 4 months it will take to implement the changes...
- How will you deal with the friction between the individual project groups and the architure team (and there will be friction?). Opportunists will take this as a wonderful opportunity to jockey for position...
- Be aware that this will be primarily a political game...

My friend, you have a tough road ahead...

The **first** step is to be crystal clear on what the architecture team is supposed to achieve.
Why are you putting the team in place?
Are you trying to unify all the applications, develop a common framework, what?
What is the mandate and the vision for this team?

Whoever the lead on this team better have kick a** interpersonal skills.
It should **not** be the brilliant coder that can whistle the star wars theme song and make light saber noises... but he should probably be on the team in a technical capacity.

You should probably populate the team with people that are familiar with the majority of the projects. I would be wary of selecting *all* the current leads, as that might take

a big chunk of knowledge from the current teams. And let's face it, those teams have to be productive while the architecture team comes up with its own deliverables.

Share Improve this answer Follow

edited Sep 23, 2008 at 19:28

answered Sep 23, 2008 at 18:59

Benoit

38.9k ● 24 ● 85 ● 117

2

Architecture is difficult to get right.

The "Architects" need the power to get things done, but need to be savvy enough to not abuse that power and completely alienate the rest of the company.

I've worked at two places where architecture teams were implemented -- one was a success, the other isn't looking so good. At the successful place, it was a relatively small environment where the head architect was recognized as a technical leader, and the other members of the team had excellent writing and political skills. Everyone acted in the best interest of the organization.

At the place that it didn't work out so well, the architects clearly represented specific factions in the organization, and didn't earn the trust or respect of the entire place. The result was that more time was spent cooking up excuses to bypass the architecture than gleaning any

value from it. In this case, frustration turned into passive/aggressive and other anti-social behaviors.

I think the other questions that you ask about scope/responsibilities/transition are all answered by "it depends". It depends on the company, the people, the money and the schedule.

Share Improve this answer

Follow

answered Sep 23, 2008 at 19:47

duffbeer703

308 ● 1 ● 9

1

Interesting question.

First, you have to have a clear notion of what problem this "architecture" team is solving. If you can not clearly define the "mission" of the team it will fail and do it with great big explosions. :)

That being said, the first step is define the problem you are solving. Are you trying to keep up with technology? Are you trying to incorporate some code reuse between projects? Are you trying to utilize your development staff to the best possible effect? There are several reasons to implement an architecture team and given your setup, any one of these might be sufficent. From your question, it looks like your goal is reworking the existing apps so that is a good first step.

Since you already have a group of leads that have good specific knowledge of the apps it would be a good idea to

start with them. Get them together and hash out what the new global architecture should look like. You might also want to get a consultant to help facilitate the conversation at this point. Define the goals of the rework and come out with a "big picture" that everybody can agree to.

After that I would take a handful of the leads and promote them (backfilling the leads from the developer pool) to the architecture team. They will then meet with the leads to ensure things are going according to that "Big Picture".

I would NOT bring in a whole new group from the outside. That would create an unwanted Us vs. Them dynamic that is never good. The outsiders would also have no idea of how things are supposed to work or why things don't work the way logic would imply they should. :)

Share Improve this answer

Follow

answered Sep 23, 2008 at 19:04

Craig

11.9k ● 13 ● 45 ● 62

1

"Architecture" in this context in itself means nothing. It means "experts on transversal topics".

Whenever you have an "Architecture team", you will have a transversal team which will provide services for many projects.

As stated by the previous answers, you need to know what topics such an "Architecture department" will have to address.

Now, here is a example of organization of architecture teams based on several topics:

- Business and Functional Architecture team: writes many business-related specifications, and checks the alignment between existing application and functional workflow, in order to complete a coherent cartography of application.
- Application Architecture team: provides the cartography, but also decide of how the functional specifications decided by the Business and Functional Architecture Team will be organized into applications.
  For example, you need a functional module for "portfolio process", but the Application Architecture team can decide to split that into a "launcher", a "dispatcher", a GUI, and so on.
- Technical Architecture teams, always composed of:
  - Execution Architecture team, for all non-business-purely-technical topics (logging, KPI, frameworks, ...)
  - Development Architecture team (tool evaluation and support, technological survey, repositories management for version and configuration control)
  - OA (Operational Architecture) for making an environment "executable" (that is, knowing the right processes, the right servers and the right

networks in order to deploy your system either for homologation or for production.)

You may want to add a Logistic team for all the management of server and networks, with the tasks of Backup and DRP strategies. And a support strategy based on a good case system.

And you are good to go.

Now, do not forget that when you begin some "large rework", your Functional Architecture will have the mission to enforce the **coherencies** both between:

- the reworked projects to be sure they stay within the fixed functional perimeter
- the **legacy** projects to be sure their maintenances do not involve *opposite choices* compared to the one applied to the reworked projects.

Any rework in a shop this size means indeed to be able to make *necessary evolutions to legacy projects* while waiting for the rework to produce the first releases. (The legacy can not just wait and stay still during 2-3 years of rework)

A large rework should involve three major milestone:

- 1/ dialog with the legacy
- 2/ complete the legacy
- 3/ replace the legacy

Meaning any given component is in effect developed three time! ;)

Good luck and good night.

Share Improve this answer Follow

edited Sep 24, 2008 at 13:47

answered Sep 23, 2008 at 19:12

VonC

1.3m ● 558 ● 4.7k ● 5.6k

0

in general, be very careful about the incentives both political and otherwise associated with the architecture group. it is far to easy for the 'architectural review board' (or whatever you want to call it) to become barriers to progress. All it takes is zero incentive to improve things and a negative incentive when things change and don't immediately improve.

realize that mistakes will be made, some 'great new technologies' will turn out to be half-baked fads, and ecourage change and innovation. this may yield the occasional short-term upheaval and failed transformation, but that is better than stagnation.

and the alternative inevitably yields stagnation; in larger organizations i have seen careers ruined because a manager believed enough in his team to support their recommendation for new technology all the way to the top, provide the case studies to prove it, and back it to the

hilt. When the new tech was finally approved (after almost a year of political infighting) the CTO (who opposed it the entire time) claimed credit for the innovation and transferred the manager to a backwoods department. In another incident a new technology was proposed, with numerous examples of success in the same business area, and a committe was formed to study the issue. *Five years later* they are still studying the issue, and nothing has been done

Share Improve this answer

Follow

answered Sep 23, 2008 at 19:10

Steven A. Lowe

61.1k ● 19 ● 135 ● 204

0

I think the architecture team needs to have people senior enough that they knows the inner workings of all the development teams and be able to say No to requests/guide lines. I have been in teams with good developers, yet don't have enough authority and ended up following whatever the higher rank developer managers from different teams and produced inconsistent frameworks.

Share Improve this answer

Follow

answered Sep 23, 2008 at 19:15

Jon

263 ● 4 ● 10

You need work through the business scenarios A) and B)

0

A) What if you don't set it up, i.e. do nothing:
Estimates of the rework on going maintenance costs

B) You do set up up then:
Disruption to near term deliverables, due to resource diversion.
Risk multiple products may be dis-avantanged in the short term.
Costs of perceived extra manpower.
Who will flag up if the products get weakened by the exercise (performance or perceived inflexibility)

Next get the product teams to so same exercise, compare results.

If you do justify it, here are two routes I have seen:
1. Pick a lead product to drive the architecture and add resources to this project.
Then be prepared to add more resources, and be patient otherwise the lead product suffers.
You risk division with this route, it worked when the lead product was 40% of revenue.
2. Start up a small team, drawn from the most promising discussions that have been occurring internally, wrap in the new architecture in each product incrementally.
Weave this teams work into the the products work.

Some Question for you to look at:
1. How soon do you have to achieve architecture convergence to get the business benefit.
2. Who are the team members already talking about architecture convergence, and are they

asking/suggesting its importance, you need this question to on the "back burner" for 80% of your team leads.

What not to do

* Hire in experts from outside (unless you are in a real mess now)
* Give up after a few months, this is long term.
* Change all projects at once.
* Start until you have a core of three that can make this happen.
* Let the architecture department become any bigger than it has to be
* Let it be perceived the architecture department will solve product teams problems
* Let any product appear to be "waiting for the new architecture"
* Let the architecture department "define all" or have scope creep
* Force all products into architecture, some may not fit (e.g. not developed in same country)

What do do:

* Armed with good justification from first question get senior management to buy in and be asking the product teams to report progress
* Make step by step changes in alignment of product to architecture map
* Work on alignment the most promising or low risk product lines first
* Setup up metrics so you can demonstrate the value add (see justification from fist set of questions)

* Have a road map for all products to get converged or not.
* Think what the core architecture delivers and who maintains its artefacts
* Allow product teams to contribute to the core in terms of specificatios, code and maintenance of core
* Setup trainig on how to use the work of the architecture team for new starters and existing teams

Share Improve this answer Follow

edited Sep 24, 2008 at 7:27

answered Sep 23, 2008 at 20:37

ICW

111 ● 1 ● 3

0

Architecture alone might turn people into astronauts / zombies. So they should definitely have some coding to do even if that's basic prototyping. In fact the success of their prototypes must be definite review factor.

They should give out by-monthly presentations / frequent blogs that track their work, so that others in the organization can learns.

There should be academic goals like being familiar with certain platforms / tools / books and design philosophies.

They should be given time to pursue new tools / projects / responsibilities in existing projects if they feel like it.

They would have the responsibility to do at-least 3-4 code reviews of critical modules and come up with code style guidelines.

They would have the responsibility to review low level designs at -least of key modules.

They should be given spare time as individuals or team to build something they feel might be useful.

They should have the option to forego architecture and return to regular work if they feel like it with no penalties involved.

They should have the information about whats happening across all projects running in the organization. At least one project should be followed closely so that they can inform their own peers about stuff happening elsewhere. This can possibly be the project in which they perform code reviews and such.

They should have a highly technical person as their manager.

Architects should be switched between projects once they are very familiar with one of them, and allowed to follow on whatever prototype they where following while working with the original project.

Have at least one real goal (Like consolidate all commonalities across projects into a single library) every 1 year

Invest time and training to ensure that architects do not get ego bound and conduct fairly professionally. Conflict resolution and other soft skills training along with budget for technical meetings and trainings would be definitely required too.

Share Improve this answer Follow

edited Sep 26, 2008 at 3:50

answered Sep 25, 2008 at 10:36

computinglife

4,391 ● 1 ● 23 ● 18