Social Dimensions to Architecting Software Systems

Rajesh Uttangi

Overview

The software development process is intense in terms of human efforts, collaboration and lack of visibility of progress on different fronts. This gives rise to quite complex and interesting social dimensions to the whole process. There is a need to look at these dynamics from a fresh perspective relative to the new enterprise development practices and paradigms.

This paper looks at the process of coming up with a Software Architecture as an area where multiple roles in the team come together, contribute and even contradict each other on many aspects. It highlights the need for social skills on the part of the Architect and the need to involve many other roles in the process. It mentions some patterns for alleviating friction and discontentment during the entire development process which fundamentally revolves around the Architecture.

Introduction

Software has the unique feature of being constructible and changeable at any point of time with relative ease, compared to the engineering disciplines like civil or automobile engineering. Think of being able to change the foundation plan for a building after it has been constructed, because it doesn't give the functionality required! But that is exactly what frequently happens to software systems - at least in parts.

Also this structure of the software can be built, torn apart, changed and recomposed again by different individuals at different times. This can lead to lack of accountability and hence make the development process hard to manage. Particularly in a long term project, from the start to completion, there may be many a change in guard.

In some cases this change in line may trigger change in the very structure of the system being built. The parties which are adversely affected by these changes are certain to feel frustrated, and there may prevail a feeling of detachment from the final destiny of the system under construction. Everybody feels things are not in their control. We examine the issues that surround these social dynamics and identify areas that need to be addressed to make the process smoother and more effective in the long run.

The State of the Industry

IT Services industry in India is yet to attain maturity in terms of definition of roles and responsibilities within the development teams. Some of the reasons for this are:

- The small size of the teams/projects being executed
- The need for flexibility
- Diverse technology mix

- The size of the companies itself.

But, the identification of roles becomes indispensable in many contexts, one being the very nature of Software development and maintenance – specifically the tendency to change frequently and unpredictably.

In terms of accountability and traceability with respect to cost, effort and timelines, it has been observed that there is a clear gap in thinking between management, and the execution teams. This is evident in what activities like maintenance and continuous integration mean.

Budgets are planned for the new features being added, but the cost of integration of these new features or modules into the existing system, which is considerably high, frequently goes unbudgeted. This leaves management thinking that as one progresses through the project development and maintenance, cost per new feature seems to keep going up disproportionately.

Other types of problems are the lack of enthusiasm among the designers and developers to follow some of the decisions, processes and guidelines laid out by the Architects and the management. This leads to frequent quality and integration problems.

Other types of problems are the lack of enthusiasm among the designers and developers to follow some of the decisions, processes and guidelines laid out by the Architects and the management. This leads to frequent quality and integration problems.

There are even developers who nurture discontent and hidden agendas against the architects and managers because they feel used, isolated or suppressed. Most often it has very undesirable outcomes, including production of work items which look nothing like what was planned, and an overworked and wasted workforce, which seems to be always on its toes to move out of the current project.

Project Management Questions

We frequently find questions like the below being asked in project management circles and no conclusive answers being provided:

- How to avoid conflicts between the customer facing team and the off-shore implementation team pertaining to the change requests accepted and the risk introduced thereof?
- How do I convert developers into stake-holders in the project with a conscious buy-in when they seem to feel that they are just responsible for churning out code and that an integrated and satisfactorily deployed system is not at all in their hands?
- How do I ensure overall cohesiveness and mutual support in the development organization?
- How do I ensure Bredemeyer's 3 C's – "Capability, Commitment and Culture" work in my team?
- How do I avoid "Kludge" (Webster's dictionary – "A system, especially a computer system, that is constituted of poorly matched elements or of elements originally intended for other applications")?

Divide between Developers and Managers

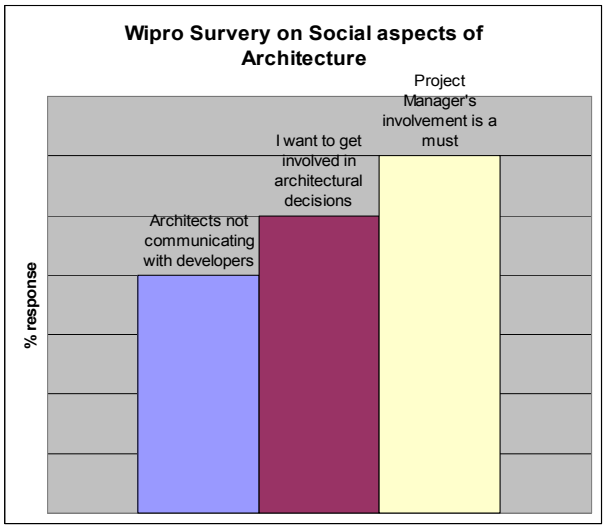
One aspect of this problem is the divide between those classified as developers and the others, called managers. It is also apparent that the team dynamics surrounding pure management roles like project managers and business managers are quite different from the impact created by a role like that of a technical architect.

The former are known to be having their own responsibilities and issues concerning finance, customer relations or other general management aspects. But the latter is seen as someone who knows the technical nature of the work being done at all levels. Often, wasted efforts and major changes are blamed on him. The notion is that because something affects the architecture, it is important and has to be done on priority and that if the architects had been careful enough, much of the additional effort could have been avoided. So there is a gap between what really caused the rework and what is perceived as the source of the problem.

There is compelling evidence that this situation is aggravated to a great extent because the social dimensions of the work place are not understood and managed. Unfortunately, human interaction dynamics and soft skills are not the primary parts of education for an architect anyway.

The Wipro Story

In a survey conducted across projects at Wipro Technologies, many developers felt that:



- Architects can do well to communicate more with the designers and more so with the developers. The implications and additional efforts due to many a decision at the higher level does not get justified and elaborated at the developers' level.
- Some developers felt that they could have contributed to the architecture of the system with significant inputs based on their experience and expertise. There was no process in place for capturing and implementing these inputs.
- Asked whether supporting roles like the project manager should or should not be part of technical decisions in the project, an overwhelming majority felt that there can be significant contributions from this group too. This seems to represent a general feeling that a majority of the development community feels their (and their peers') estrangement from important issues and aspects like the architecture.

Defining Software Architecture

But what really does architecture mean to a cross section of the development teams? A standard and uniform definition for software architecture is elusive. Some

interesting definitions convey that architecture is everything about the project that the development team thinks as really important. Conversely, what is not important to the developers does not qualify as architecture.

By these definitions, coding would be part of architecture and by some others it would not. Who then decides what is important? We can see various primary players and many undercurrent players who influence this choice and stress the need to get started with the thing. It is more often the economic, social and political considerations that guide the decisions. Or at least, the decisions have a significant social flavour attached because the team is embarking on a journey into unchartered waters, and common sense and business judgments significantly determine what course to take.

Structure of the Development Organisation

In this scenario what matters is the structure of the development organization and the processes in place. Every team member has to have faith in the decisions made by the managers and the architects, and the architects in turn become the starting point of any technical activity that takes place.

At this stage, as the uncertainty in the atmosphere is more than apparent, many senior developers may harbour notions like they know the solution to the problem being addressed better than those who call themselves architects. Unless the situation is managed and an ambience of openness and confidence is built, things could go haywire. This really highlights the need for effective communication and showing that the Architect is really the torch bearer, a reliable leader, or more so, a mentor and friend.

This seems to represent a general feeling that a majority of the development community feels their (and their peers') estrangement from important issues and aspects like the architecture.

Conway observed that there is a strong relationship between the software created in an organization and both the structure of that organization, and how the people in that organization really communicate. To put this insight into practice, you need to use not just the formally documented structure and processes of your organization, but you also need to observe and learn the informal communication paths as well.

Alistair Cockburn in a paper titled 'On the Interaction of Social Issues and Software Architecture' says "It has long been said, "Architecture follows Organization" and "Organization follows Architecture"…. Architects do not like being told their clean design is a result of accounting for social forces. Project managers do not use their knowledge of social issues to influence architecture. Yet it is clear that social issues affect the software architecture in ways that the good architect takes into account".

Architecture Has to Reflect the Interests of the Stakeholders

Throughout the development cycle, architecture involves many aspects – business strategy, technical infrastructure, competitiveness, data and above all, delivering value to the stakeholders like the users, developers, managers and the architecture team itself. While the interests of the user community are well understood and religiously protected, the other stakeholders don't find so much support from the architecture. Because still they are expected to work for realizing the architecture and not use it as a tool to come up with a good system, bringing their own expertise to bear.

Architecture Representing All The Organisation

The architects on the team represent knowledge of the products of their organizational group as well as the relative priorities of their system requirements. Architects cannot exist and work in isolation from the rest of the organization.

In complex systems, no one person covers the full breadth of technical expertise required to properly inform the architectural decisions, and lend credibility to the architecture. The system definition should represent the interest of different organizations. The architects on the team represent knowledge of the products of their organizational group (project, division, etc.) as well as the relative priorities of their system requirements (organizational goals, and product functionality and qualities). Architects cannot exist and work in isolation from the rest of the organization.

Speaking of the architect as a creator of the foundation on which the team is going to bet its many months of effort and time, Martin Fowler talks about two species of architects, the first being Architectus Reloadus. He is a person who makes all the important decisions about the project, because a single mind is needed to ensure a system's conceptual integrity, and perhaps the architect doesn't think the team members are sufficiently skilled to make those decisions.

A Common View

But frequently the challenge for the architecture team is not to create an architecture that is "as if of one mind"- that is, having the quality of integrity - but to create one at all. Many times it is even difficult to be able to orchestrate the minimal conditions required to complete the job. The reasons for settling for such a substandard work product are the difficulties on the social front - being able to sell the "vision" to the team, acceptable distribution of roles and responsibilities, common levels of understanding about what is to be done, timely communication and separation of concerns, etc. The trickiest ones are common understanding and communication.

Philippe Kruchten says that "the practice of architecture is a long and rapid succession of sub-optimal decisions, mostly made in partial light." Architecture projects are, by their very nature, ventures into unchartered territory and especially fraught with competing ideas on which direction to take. Without obtaining a consensus and solid leadership, indecision is like to reign.

Problem with Part Time Architects

Architecture teams made up of part-timers, typically fail to gain traction on the problem. Faced with competing short-term product pressures, the architecture effort stalls. This may make even very trivial things like organizing and conducting meetings quite difficult. Frequently, critical viewpoints are not represented when a decision is made, resulting in lack of buy-in to the decision. When management finds that the indecision is more due to internal conflicts, becomes impatient with slow progress, and sees no quick results coming out of the 'Kludge', the entire effort can fail.

Definition of Roles helps but not sufficient

Many organizations have a weak or confused notion of the responsibilities and interaction of roles within the organization.

As regards to roles, the story doesn't get any better. Teams are in a constant state of flux and who should be doing what is always a point of conflict. Many organizations have a weak or confused notion of the responsibilities and interaction of roles within the organization. In such conditions architecture teams may cut themselves off from the rest of the organization, produce an architecture, but it will be in great danger of being rejected.

Avoiding Architectural Isolation

So what should one do to avoid such a discord and chaos? The key might be in avoiding temptations to try to reduce distractions, putting up a "wall" around the Architectural team so that they can make rapid and efficient progress.

This isolation is the source of serious misunderstandings. Firstly, the architecture team may come to be resented as a "select" group off doing interesting stuff that is disconnected from everyday product pressures. In an atmosphere of resentment, the developer community will be looking for opportunities to find weakness in the architecture.

Manage Priorities and Concerns

Secondly, keep track of product priorities and developer concerns. Unless there is strong communication between the architecture team and developers and project managers on the one hand, and strategic managers on the other, the system cannot be planned and executed as one cohesive unit.

A good Architect is represented by Fowler's second archtype - Architectus Oryzus. This is an architect who is very aware of what's going on in the project, looking out for important issues and tackling them before they become serious problems. He is more suitably a guide and mentor who is simply a more experienced and skilful team member who teaches the other team members to better fend for themselves, yet is always there for the really tricky stuff.

When one sees an architect like this the most important and noticeable part of the work is intense collaboration. This leads to the rule of thumb that an architect's value is inversely proportional to the number of decisions he makes himself (vs. leveraging on the team to make those decisions). The architect cannot limit himself to drawing high level architectural views of the system and hoping that it will be implemented by the developer just as the architect had in mind. More likely is the setting where in the morning the architect programs with the developer, in the afternoon participates in a requirements session, helps explain to the requirements people the technical consequences of some of their ideas in non-technical terms – such as development costs. Somewhere in between he is discussing with the project managers how the milestones need to be defined for the project.

Maintain Contact with Customers

Third, technical risks might be curtailed if the architect can maintain direct contact with well informed customers which may result in keeping the architecture effort informed of market directions. Of course this is not easily achieved. First, the architect has to gain enough confidence to be able to approach friendly customers to

discuss latest technologies. There needs to be built a solid rapport to prevent any misunderstandings and other issues.

Precursor to Starting the Architecture

One can ask a set of techno-social type of questions even before starting with the drawing board to bring every team member to the same level of understanding and build a buy-in:

- Is the level of understanding on the requirements the same across the team?
- Have all the non-functional requirements been elaborated?
- Is the idealized design of the system to be developed uniformly understood by each?
- Skills - Does the team have the proficiency required?
- Technology - Will the selected technologies work together? What are the issues with integration?
- Political - Are the fundamental goals of a project understood and agreed upon by all the relevant stakeholders in the organization?

Sharing the Architecture

The architect, being aware of the technical issues has to avoid giving an impression that he is the "only owner" of the architecture, because as things stand at this point, architecture is indeed the system itself. He can facilitate a decision but with a consciousness that he is not the only one who is going to be doing the implementation.

That there will be changes - more than a handful - in requirements is not even worth mentioning. But looking at this statement from F Brooks one really wonders; "the organization chart will initially reflect the first system design, which is almost surely not the right one … as one learns, he changes the design … Management structures also need to be changed as the system changes…". It is more than likely that there will be disappointments, resentment and tempers when one discovers that there is work being thrown away and rework is necessary. To avoid such a situation, we might take refuge in some of the patterns available to minimize the impact of change.

Some Patterns for Separation of Concerns

A set of techno-social patterns are given by Allistair Cockburn and David M. Dikel, David Kane. Cockburn says that that most software architects do not think of themselves accounting for social issues, but that is one of the characteristics of good architecture. The patterns are listed below.

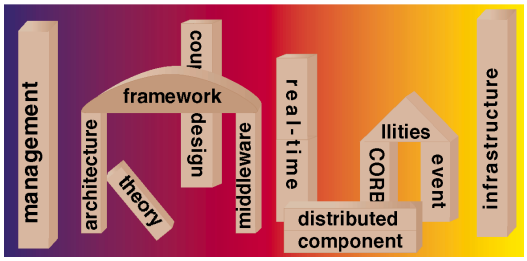
- Variation Behind Interface
- Subsystem By Skill
- Owner Per Deliverable
- Subclass Per Team
- User Interface Outside and
- Facade.

And by David M. Dikel, David Kane:

- Informal Network
- Rotation
- Airing the Laundry
- Risk Agent and
- Strong Architect

The patterns attempt at converting each team member into a stakeholder in the architecture and by providing separation of concerns and minimal impact. They intend to bring the team into a co-operative mode with respect to the efforts required as a result of requirement and implementation changes. Some of the patterns ensure work unit delivery issues.

Process Framework



In addition to the above, having a process framework, which is an industry standard, easy to understand, to follow, and is easily accessible, might be used to determine the roles, responsibilities and deliverables in a project. It provides a structure and a reference model for everybody to operate within and prevents responsibility overlaps, passing-the-buck syndrome and confusion. Rational Unified Process is one of the popular frameworks available.

Vision, Communication and Organization

Finally, it all boils down to vision, communication and organization. The key purpose of architecture is to facilitate team communication and understanding. In software development methodologies avoid risk by encouraging communication between various roles in a project.

Finally, it all boils down to vision, communication and organization. The key purpose of architecture is to facilitate team communication and understanding. In software development methodologies avoid risk by encouraging communication between various roles in a project.

At the highest level, the software architect will work with the sponsor to identify architectural requirements while the business analyst identifies the business requirements. Together they will model the process the software must fulfill, and the means by which it will do so. The sponsor will then identify those who have input into the project, and the model grows as it moves down through the ranks.

Detail is added as appropriate, and when the model is presented to development, the architectural process moves from one of vision and design to one of management. In organizational structures, a software architect will begin to layer development efforts. A team of a hundred will have a senior software architect, and further software architects who lead individual projects or manage specific areas of expertise. Senior developers will take the software architects' models and convert them to skeleton code for junior developers and specialists to complete. Doing so will make development run more smoothly, be more controlled and more measurable.

In essence, architecting function is ideally performed by every team member who feels a part of the solution, owns his responsibilities willingly because he/she has helped define it and can stand accountable for problems arising in context of – now not so esoteric – a thing called Architecture.

Conclusion

While there is hardly any need to highlight the Technical issues associated with Architecture, the social aspects that continue to have ever increasing influence cannot be ignored. There is no doubt that there are team members who feel used, left out or till treated in almost all projects, particularly when the question of their involvement in the system's architecture comes.

The problems are exacerbated by ever growing size of teams, integration of cultures, and faster turn over rates, increasing complexity and cross technology interoperability concerns. If issues of stakeholder involvement, assimilation of ideas, recognizing contributions and mutual support issues are systematically, yet informally addressed, results can be pleasantly surprising.

References

1. David Dikel, David Kane, Conway's Law Revisited: Successfully Aligning Enterprise Architecture, May 01, 2002

2. David M. Dikel, David Kane, and James R. Wilson, *Software Architecture: Organizational Principles and Patterns,* Prentice-Hall, 2001

3. F. Brooks, *The Mythical Man Month, Anniversary Ed.*, 1995

4. Marti Fowler, Who needs an architect? IEEE Software, Sept/Oct 03 edition, pp 11,12

5. Software Architecture as a shared mental model, Ric Holt, Position of paper[2001]

About the Author

Rajesh Uttangi is a Specialist with the Rational Competency focus group, Talent Transformation, Wipro Technologies. His main areas of interest include OOAD, UML, RUP, Software Architecture and J2EE. He can be reached at rajesh.uttangi@wipro.com

Project Perfect is a project management software consulting and training organisation based in Sydney Australia. Their focus is to provide creative yet pragmatic solutions to Project Management issues.

Project Perfect sell "Project Administrator" software, which is a tool to assist organisations better manage project risks, issues, budgets, scope, documentation planning and scheduling. They also created a technique for gathering requirements called "Method H"™, and sell software to support the technique. For more information on Project tools or Project Management visit www.projectperfect.com.au