SEMAT position paper

Challenges we are facing today

Software engineering is a very large area. It is not possible to find one process that covers the whole area and at the same time is reasonably simple to use. The process would have to cover all combinations of small and large projects, hard and soft real time and non real time Challengers, small and large data volumes and different degrees of distribution, reliability, security and so on. Either it has to be a very large and complex process which includes large parts of software engineering literature or many separate processes, each one for a different problem area. There is either no obvious way how all relevant information should be organized in a component based framework to make it smooth to compare, split, combine and merge components that would result in a ready-to-use process for a certain environment such as small project, hard real time, large data volume, undistributed, high security .

We will describe some of the central challenges with processes and software engineering in large projects in the embedded area. The projects in this context may contain hundreds or thousands of developers. The development of the first release will take a long time and the system will then be significantly enhanced during decades.

In the startup of a new big project people are gathered from different old projects and start working with setting up a new technical platform and new processes. Of course we would like to select the successful parts from previous platforms and combine these with new parts. The same we would like to do with processes.

It is however very hard to evaluate and merge different processes.

- People with different backgrounds have a hard time understanding processes that they have not used in practice. Different processes use different terms to refer to basically the same concept, and terms are poorly defined. This creates misunderstandings which take a long time to resolve.

- It is hard to pick parts out of a process, since there are no clean interfaces to the surrounding parts. If you attempt to take out a part you will get a ripple effect and have a hard time not ripping out the whole process.

- When a new process is introduced the users often think that concepts in a new process are totally new and have a hard time relating them to similar concepts in other processes.

- Instead of making real improvements in productivity and quality we are tempted to follow new processes that seem to provide new ways of doing things but are merely the same process in new clothes.

All this leads to confusion and unproductive discussions that can take a long time.

- Often the consequence will be to restart more or less from scratch. The good insights are then lost.

- Even when there is no start from scratch good parts from some processes are thrown away and less good parts are chosen.

- There will be a significant time before the process is really productive. The merges will take a while and there is no common understanding of the process until it has been practiced for a period.

Instead of a continuous flow over the years where we improve all the time we get interrupts where we start more or less over from scratch. This is a waste of knowledge and resources and even more serious – it prevents us from improving in the long run. So much effort is spent in the restarts that we do not get time for real improvements.

Such restarts do not only happen at large project startups – they can happen at any point where new technology or methodology is introduced.

There are similar challenges in process improvement work.

The good insights that emerge during a project are packaged into a new process. This process then contains modified versions of existing parts that did not work that well. But since the new process is a separate document the relation to previous processes is not explicit and is lost.

When several such processes have been developed the problem emerges. Which process should be used? They all contain good advice but it is not clear how they should be combined. These conflicts are hard to sort out, we would like to pick the improvements we think are relevant for a given project. How nice it would be to be able to keep the original part (as an independent component) and then add the modifications as separate independent components with a specialization relation from modifications to originals. Then one would be able to pick the relevant improvements for a certain project.

What is successful?

Projects gradually come to good insights what is working and not. Processes are improved and after the first release things in general works well. Developers know what to do and how to get the right information and deliver results. Planned release dates can be met and requirements on performance and quality can be achieved.

What is needed?

We need a flexible framework to build ready-to-use processes with. The framework should contain:

- Fundamental process-components which are general and relevant for all processes.

- A set of dependencies and associations that can be used to describe relations between process-components and under what circumstances (project size, real time, data volumes and complexity, distribution, reliability and so on) they are applicable.

- Mechanisms that can be used to build ready-to-use process-components.

A small team should develop the fundamental process-components in order to get integrity and consistency. A lot of effort should go into defining terms as precisely as possible. This work would be analogous to the UML development where many methodologies were condensed into one.

- Organizations should be able to use the framework to define processes from scratch or to build on fundamental or other processes-components.

- Organizations should be able to package improvements to processes as process-components. This implies that process-components should be small (probably a hierarchy or network structure is needed). Otherwise would large copies of process-components be made and the precision in modification is lost.

- Organizations should be able to build new processes by picking from existing process-components – including process-improvements.