Generate Code

Español

Entries

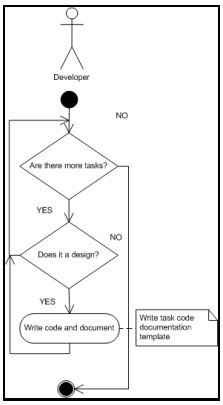
- Design Task

Exit

- Task Code
- Documentation Task Code

Solution

Process



Development time

- To acquire the necessary knowledge to develop the software product:
- To create the Product Pattern: 45 minutes.
- To apply the Product Pattern:

Explanatory video

- Not applicable

Related Patterns

- Design Pattern
- Register Progress Pattern

Quality Controllers

- None

Templates

- Codigo_Tarea.xx
- Documentación_Codigo_Tarea.doc

Examples

- None

Support Tools

- It will need a text editor such as OpenOffice Writer o Microsoft Word.
- As well as a tool Visual Paradigm for UMLfor performing exposed diagrams.

Initial Context

This product can be used in any project in which the philosophy of working in pairs to implement code is followed.

In XP the entire production software is written in pairs, two programmers sitting side by side on the same computer.

This practice ensures that the entire production code was reviewed by at least for one programmer, and generates better designs, better testing and better code. It may seem inefficient that two programmers do the "work of one programmer", but the opposite is true. Studies of pair programming show that couples produce better code in about the same time as one programmer working alone.

Result Context

Small pieces of code implemented by two people are obtained, which reduces the error rate (d).

The XP teams use a standard code in common, so that the system code looks like it was written by a single person very competent. No matter much the standard in itself: it is important that the code looks familiar, to allow collective ownership.

Problem

Pair programming is not that a person type and the other look, it is a dialogue between two people trying to simultaneously schedule, design, analyze and test.

Restricciones (*Forces*)

- **Characteristics of organizations:** This pattern can be used in existing projects in any company.
- **System Type to develop** This product can be used in projects in which user requirements are changing.
- **Type of Customer:** It must exist or be achieved, the target area development business being involved in achieving it.
- **Heuristics for use:**: If you need urgent application or dispose of some of its functionality.

Roles

- Developers (2-12)

Lessons Learned

- Pair programming is not a tutorial. If there is an expert and a novice programmer in the couple, the first should advise and guide to the other so that after a short time the novice programmer reach the required level.

Developers pair must be well organized. If it is detected that the relationship between the partners is not good, it is advisable to lose some time in reorganizing them. The most important for teamwork is the communication between members of the pair of programmers. The implemented code should be simple, for which recommendations should be followed:

1. The system (code and test) must communicate everything you want to communicate and nothing else.
2. The system must not have duplicated code
3. The system should obtain the smallest possible number of classes
4. The system should obtain the smallest possible number of methods

We must feedback of learning in design. It should be cyclically throughout the process as soon as possible:

1. Small initial investment
2. Assume simplicity
3. Incremental change
4. Traveling Light (no attempt to cover expectations not described in the design)

- It should be writing code jointly by two people working together.
- The Design Task file is a document with the design of the task to implement.
- The output product source code is the source code that implements the task.
- The document Source Code Documentation is documentation that explains the implemented code.

Capability Level

- Not applicable.

Basic Knowledge and Skills

Knowledge

- Knowledge of coding standard that defines the shared code ownership and the rules for writing and documenting code and communication between different pieces of code developed by different teams. Programmers have to follow the so that the code in the system look like if it had been written by one person.

- Knowledge of the common vision of how the program works in which the activities take place.

Abilities

- Ability to work in group. All on an XP computer contribute in any way they can.
- Predicting what will be completed by the deadline, and determining what to do next.
- Programming capability in pairs. Besides to generate better code and tests, used to communicate knowledge through teams.

Information Resources

- Álvarez, José R. y Arias Manuel. Método Extreme programming.Recuperado el 2010-03-05 de http://www.ia.uned.es/ia/asignaturas/adms/GuiaDidADMS/node61.html
- Anaya Villegas, Adrian. A proposito de programación extrema XP(extreme Programming).Recuperado el 2010-02-10 de http://www.monografias.com
- Beck, K.(2000), Una explicación de la programación extrema. Aceptar el cambio. Ed. Addisson Wesley.
- De Seta, Leonardo. Una introducción a Extreme Programming.Recuperado el 2010-03-02 de http://www.dosideas.com/noticias/metodologias/822-una-introduccion-a-extreme-programming.html
- Extreme Programming: A gentle introduction. Recuperado el 2010-03-15 de http://www.extremeprogramming.org/
- Joskowicz, José. Reglas y prácticas en Xtreme Programming. Recuperado el 2010-03-15 de http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf
- Letelier, Patricio y Panadés Mª Carmen. Metodologías Ágiles en el desarrollo de software: extreme programming. Recuperado el 2010-03-15 de http://www.willydev.net/descargas/masyxp.pdf
- Newkirk, James y Martin, Robert C.(2001), La programación Extrema en la Práctica.Ed Addisson Wesley.