UNIVERSIDAD NACIONAL DE LA PLATA
FACULTAD DE INFORMÁTICA

# TALLER DE TECNOLOGÍAS DE PRODUCCIÓN DE SOFTWARE (Opción D)

**Año 2014**

Carrera: *Analista Programador Universitario*

Año: 3°
Carácter: Obligatoria
Régimen de Cursada: Semestral
Correlativas:
Introducción a las Bases de Datos -
Algoritmos y Estructuras de Datos -
Introducción a los Sistemas Operativos -
Orientación a Objetos 1
Taller de lecto comprensión y traducción en Inglés
Ingeniería de Software 2
Profesor: *Christian Rodríguez*
Hs. semanales: 6 hs.

## FUNDAMENTACIÓN

*Dentro del marco de la carrera, esta opción propone ampliar los conocimientos con los que cuentan los alumnos abordando las metodologías ágiles para la producción de software. En su contenido, se incluyen planificación de requerimientos, técnicas de testing y programación basada en el lenguaje Ruby. Todas estas componentes se combinan en un entorno muy apropiado donde los alumnos podrán experimentar cada una de las etapas reales del desarrollo que van desde el análisis, desarrollo, prueba y puesta en producción.*

## OBJETIVOS GENERALES:

Introducir a los alumnos en un esquema de organización de producción de software, utilizando metodologías, prácticas y herramientas actualizadas y acordes con los estándares actuales.
Fomentar la práctica del alumno en esquemas de trabajo similares a los que se utilizan en las empresas de desarrollo de productos de software.
Ofrecer a los alumnos alternativas tecnológicas, siempre en base a herramientas de utilización actual en el mercado laboral.

## CONTENIDOS MINIMOS:

- Introducir un ambiente de desarrollo de software estandarizado (con herramientas integradas que den una visión homogénea y estandarizada de las aplicaciones, su interfaz grafica, el acceso a las bases de datos y la interconexión entre aplicaciones), enfocado a un organismo o "clase" de empresa usuaria.
- Practicar como usar el ambiente de desarrollo y las diferencias que tiene que con el ambiente de producción, ilustrando la metodología organizacional del pasaje de desarrollo a producción.
- Practicar con documentación estandarizada (por ej. Casos de Uso con UML)

mostrando como se pasa de una especificación al código ejecutable.

- Ejemplificar la actividad del tester de aplicaciones. Metodología de trabajo y ambiente de prueba (diferencia con los otros ambientes)
- Plantear el proceso estandarizado de desarrollo de software en una tecnologia de uso en el mercado. Rol de la documentación en cada etapa.
- Plantear el desarrollo de una solución a un problema real y que ilustre todas las problemáticas antes descriptas
- Describir cuales son las principales características de un proceso de desarrollo de software con calidad (introduciendo los principios básicos de CMM o CMMI)

**PROGRAMA ANALÍTICO**

1. *Introducción a las metodologías ágiles*
2. *Sistemas de control de versiones. Introducción a Git*
3. *Introduciendo Ruby*
4. *Máquinas virtuales de ruby: MRI, Rubinius, jRuby, etc. RVM*
5. *Mixins*
6. *Rake*
7. *TDD*
8. *Gems, rubygems, bundler*
9. *Ruby y la web*
   - *9.1 Rack*
   - *9.2 Sinatra*
   - *9.3 ORMs*
   - *9.4 Rails*

**METODOLOGÍA DE ENSEÑANZA**

*La metodología es del tipo taller, con clases teóricas donde se desarrollan los aspectos conceptuales del lenguaje, que se dictan utilizando presentaciones digitales. Se prevé dictar las clases teóricas en la sala de PC, con la asistencia de 2 ayudantes. Estas clases no son obligatorias pero se tomará asistencia.*

*Las clases prácticas se realizan en la sala de PC de la facultad, los estudiantes plantean sus dudas y trabajan con los ayudantes, quienes los acompañan en este proceso. En total se deben completar 6 prácticas. Algunas de estás prácticas incluyen ejercicios para entregar de carácter grupal. Cada entrega incluye una instancia de coloquio donde el docente a cargo del grupo realiza distintas preguntas sobre la temática abordada. La entrega se realiza a través de la plataforma virtual.*

*Esta metodología se complementa con la plataforma virtual Moodle, donde se publicará el material teórico, práctico e intercambiarán consultas por medio del foro.*

Se utilizará también una herramienta para el seguimiento de la cátedra, de la asistencia de los alumnos a las prácticas y del aprovechamiento de cada práctica.

mostrando como se pasa de una especificación al código ejecutable.

- Ejemplificar la actividad del tester de aplicaciones. Metodología de trabajo y ambiente de prueba (diferencia con los otros ambientes)
- Plantear el proceso estandarizado de desarrollo de software en una tecnología de uso en el mercado. Rol de la documentación en cada etapa.
- Plantear el desarrollo de una solución a un problema real y que ilustre todas las problemáticas antes descriptas
- Describir cuales son las principales características de un proceso de desarrollo de software con calidad (introduciendo los principios básicos de CMM o CMMI)

**PROGRAMA ANALÍTICO**

1. *Introducción a las metodologías ágiles*
2. *Sistemas de control de versiones. Introducción a Git*
3. *Introduciendo Ruby*
4. *Máquinas virtuales de ruby: MRI, Rubinius, jRuby, etc. RVM*
5. *Mixins*
6. *Rake*
7. *TDD*
8. *Gems, rubygems, bundler*
9. *Ruby y la web*
   - *9.1 Rack*
   - *9.2 Sinatra*
   - *9.3 ORMs*
   - *9.4 Rails*

**METODOLOGÍA DE ENSEÑANZA**

*La metodología es del tipo taller, con clases teóricas donde se desarrollan los aspectos conceptuales del lenguaje, que se dictan utilizando presentaciones digitales. Se prevé dictar las clases teóricas en la sala de PC, con la asistencia de 2 ayudantes. Estas clases no son obligatorias pero se tomará asistencia.*

*Las clases prácticas se realizan en la sala de PC de la facultad, los estudiantes plantean sus dudas y trabajan con los ayudantes, quienes los acompañan en este proceso. En total se deben completar 6 prácticas. Algunas de estás prácticas incluyen ejercicios para entregar de carácter grupal. Cada entrega incluye una instancia de coloquio donde el docente a cargo del grupo realiza distintas preguntas sobre la temática abordada. La entrega se realiza a través de la plataforma virtual.*

*Esta metodología se complementa con la plataforma virtual Moodle, donde se publicará el material teórico, práctico e intercambiarán consultas por medio del foro.*

Se utilizará también una herramienta para el seguimiento de la cátedra, de la asistencia de los alumnos a las prácticas y del aprovechamiento de cada práctica.



## EVALUACIÓN

*Algunos prácticos incluirán ejercicios de entrega obligatoria o evaluacion online por medio de la plataforma virtual. Las entregas formarán parte del trabajo final para la aprobación de la cursada.*

*La **aprobación de la cursada** estará dada por la aprobación del 80% de los trabajos prácticos y el trabajo final integrador.*

*La **aprobación de la materia** estará dada por la aprobación de la cursada y un extensión del trabajo final de cursada. La asistencia a las clases teóricas aportará a la calificación final.*

La nota promedio de todos los trabajos será la nota final de la materia

## BIBLIOGRAFÍA OBLIGATORIA


- H. Kniberg, Scrum and XP from the Trenches, InfoQ
- K. Schwaber/M. Beedle , Agile Software Development with Scrum
- E. Gamma/R. Helm/R. Johnson/J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley
- D. Thomas, Programming Ruby 1.9, The Pragmatic Programmers' Guide
- G. Brown, Ruby Best Practices, O'Reilly
- S.Ruby/D. Thomas/D. Hansson, Agile Web Development with Rails, The Pragmatic Programmers' Guide
- A. Harris/K. Haase, Sinatra: up and running, O'Reilly


## BIBLIOGRAFÍA COMPLEMENTARIA


- Scrum in five minutes, Softhouse
- Rails Guides: http://guides.rubyonrails.org/
- Ruby doc: http://www.ruby-doc.org/
- Try Ruby: http://tryruby.org

**UNIVERSIDAD NACIONAL DE LA PLATA**
**FACULTAD DE INFORMÁTICA**

**CRONOGRAMA DE CLASES Y EVALUACIONES**
**Fecha de comienzo 17/08/2013**

| Clase | Contenidos/Actividades | Evaluaciones previstas |
|---|---|---|
| 1 | Metodologías ágiles<br>Sistemas de control de versiones<br>Introducción a GIT | |
| 2 | Uso de GIT.<br>GitHub<br>Introduciendo Ruby | Evaluación online sobre conceptos de metodologías ágiles |
| 3 | VMs Ruby.<br>Instalación del entorno<br>Objetos, arreglos, hashes, símbolos<br>Estructuras de control | Evaluación coloquial del uso de GIT |
| 4 | Bloques e iteradores<br>IO estándar<br>Clases, objetos, métodos y variables | |
| 5 | Módulos<br>Mixins<br>Tipos estándar | |
| 6 | Rake<br>TDD | Entrega de un desarrollo Ruby |
| 7 | Gems / Rubygems<br>Bundler | Entrega de un desarrollo Ruby utilizando TDD |
| 8 | Rack | |
| 9 | Sinatra framework | |
| 10 | ORMs | |
| 12 | Rails framework | Entrega de aplicación web sinatra |
| 13 | Rails framework | |
| 14 | Rails framework | |
| 15<br>16<br>17<br>18 | Desarrollo de trabajo final | Entrega de aplicación web rails |

**Contacto de la cátedra (mail, página, plataforma virtual de gestión de cursos):**
car@info.unlp.edu.ar
http://catedras.info.unlp.edu.ar

**Firmas del/los profesores responsables:**

Rafael Cristia