MINISTERIO de EDUCACIÓN CIENCIA y TECNOLOGÍA PRESIDENCIA de la NACIÓN

Instituto Nacional de Educación Tecnológica

Perfil Profesional

Técnico en Programación

Técnico de Nivel Medio en Programación de Computadores

Ámbito de Desempeño

A partir de especificaciones de diseño de software, los programadores escriben, verifican y arreglan instrucciones detalladas que configuran programas de computación. La ejecución de éstos en equipos de computación produce los resultados que constituyen los objetivos del sistema. También conciben, diseñan y verifican estructuras lógicas para resolver problemas mediante computadores. Estos programadores pueden provenir de diversos niveles de educación formal o informal, aunque buena parte de ellos proviene de educación superior incompleta, y a veces se los identifica por la tecnología que emplean. Una formación posible en el marco de la educación media es la de Técnicos en Programación de Computadores.

Hay organizaciones de diversos tipos que ocupan a programadores. Empresas que realizan desarrollo de software por encargo de otras organizaciones locales o extranjeras, que proveen software junto con otros servicios de asesoramiento y consultoría, y, en menor número, que desarrollan sus propios productos de software para vender en el país o en el exterior, y organizaciones dedicadas a otras actividades, pero que producen el software que necesitan para desarrollar sus propias actividades o que integran en productos que venden.

El Técnico en Programación de Computadores participa en proyectos de desarrollo de software desempeñando roles que tienen por objeto producir programas, módulos o componentes de sistemas de computación. Estos módulos suelen integrarse en aplicaciones o subsistemas que interactúan entre si, con otras aplicaciones ya existentes desarrolladas con la misma o distinta tecnología, con el sistema operativo del computador u otro software de base (motor de base de datos, navegador, monitor de comunicaciones) configurando distintas capas de software que pueden estar distribuidas en diversas máquinas situadas en la misma o distintas ubicaciones.

Típicamente, el proceso de desarrollo de software es una tarea grupal que adopta la forma de proyecto, el que es negociado y acordado con el cliente o usuario por un Gerente y conducido y administrado por un Líder responsable operativamente por el mismo. El equipo de trabajo se compone de este Líder; uno o más Analistas Funcionales que interactúan con los usuarios para obtener y especificar los requisitos que debe cumplir el sistema (aunque a veces los requisitos vienen dados y no hay interacción con el usuario); un Arquitecto o Diseñador de Software que establece el diseño general y especificaciones de la solución; uno o más Analistas Técnicos que se ocupan de detalles relativos a la tecnología, las bases de datos o los estándares de programación y asesoran y dan apoyo técnico a los programadores.

El rol más numeroso en el equipo del proyecto es el de los desarrolladores o programadores, que reciben las especificaciones de diseño o requisito de modificación del Líder o alguien que los coordine y, después de haber construido o modificado y verificado el módulo asignado, lo entregan a un responsable de configuraciones que registra y administra las versiones, dando intervención a un grupo de *testing* (verificación del cumplimiento de las especificaciones del software, segundo en cantidad de gente) que lo integra con el trabajo de otros programadores y lo somete a pruebas para verificar su funcionalidad y calidad, y lo aprueba o devuelve con observaciones para, finalmente, liberarlo para su instalación y uso.

Perfil Profesional

El Técnico en Programación de Computadores estará capacitado para realizar programas o componentes de sistemas de computación – interpretar especificaciones de diseño, documentar los productos realizados, verificar los componentes programados, buscar causas de malfuncionamiento y corregir los programas o adaptarlos a cambios en las especificaciones – desarrollando las actividades descriptas en el perfil profesional y cumpliendo con los criterios de realización establecidos para las mismas en el marco de un equipo de trabajo organizado por proyecto.

Áreas de Actividad

1. Interpretar, en el contexto del proyecto, **especificaciones de diseño** o requisitos de las asignaciones a programar. Comprende validar la coherencia e integridad de las mismas y convalidar su propia interpretación con quienes la hayan realizado o provisto.

2. Planificar su propio trabajo en el contexto del equipo de desarrollo del proyecto.
Implica identificar aspectos de posible dificultad o riesgo que requieran consulta o un cuidado o actividad extraordinarios, evaluar a priori la magnitud del esfuerzo requerido para lograr la solución del problema, anticipar la posibilidad de cumplir en tiempo y forma con lo requerido, considerar la posibilidad de subdividir la asignación en pasos o componentes menores y establecer informalmente un orden o secuencia de trabajo, así como gestionar el entorno que se requiere para afrontar el desarrollo de la asignación recibida.

3. Analizar estrategias para desarrollar la asignación recibida en el contexto del proyecto y de la tecnología a utilizar. Implica investigar para refinar aspectos ambiguos o insuficientemente conocidos del diseño o los requisitos de la solución, resolver problemas de lógica que implican diseño o refinación de algoritmos o estructuras de datos que faciliten o permitan la solución, buscar componentes disponibles y adecuados para utilizar en la solución, bosquejar eventuales estrategias alternativas y evaluarlas para seleccionar la más apropiada.

4. Producir el código que resolverá la asignación en el contexto de la tecnología asignada al proyecto. Esto incluye la modificación (agregado, reemplazo o eliminación) de código ya escrito, sea para corregir errores observados en pruebas o para cambiar funcionalidades o el comportamiento de productos con existencia previa.
Comprende la definición o instanciación de clases, escritura de algoritmos, estructuración de datos necesarios, o la incorporación y eventual adaptación de componentes obtenidos de bibliotecas o de otros programas, respetando estándares de buena práctica y normas internas de la empresa o proyecto, así como identificando componentes o partes que puedan ser potencialmente reutilizados en el futuro.

5. Verificar unitariamente el producto desarrollado para asegurarse que cumple con las especificaciones recibidas. Implica planificar y documentar casos de prueba, preparar datos y entornos de prueba (“testing”), generar código adicional para simular el entorno o activar las pruebas, analizar causas de comportamientos o resultados no previstos para corregir el código incorrecto o preocuparse por mejorar la eficiencia (“tunning”) de la solución. También comprende participar en la realización y control de pruebas de productos de otros.

6. Depurar el código de programas para decidir qué hay que corregir. Implica revisar especificaciones y código de componentes unitarios para encontrar las partes o instrucciones que provocaron malfuncionamientos, incidentes reportados o ineficiencias, con el objeto de analizar sus causas y definir acciones correctivas.

7. Realizar, con otros programadores o con especialistas, **revisiones cruzadas de código o de interfaces**. Implica revisar el cumplimiento de especificaciones, de estándares y de buenas prácticas, evaluar el uso eficiente de recursos y del ambiente de desarrollo y aportar observaciones con propuestas de cambios tendientes a mejorar la calidad, mantenibilidad y eficiencia del producto.

8. Documentar su trabajo para que resulte interpretable y utilizable por quienes lo necesiten. Comprende comentar en línea el código y las clases, complementar los documentos de diseño, aportar soluciones, códigos y consideraciones a una base común de conocimiento, confeccionar o completar los reportes, entre ellos los de incidentes, requeridos y adjuntar resultados de las pruebas o advertencias sobre posibles limitaciones de la solución.

Desempeño de base

9. Explotar las funcionalidades de los sistemas informáticos para la realización de sus actividades**.** Implica conocer y saber utilizar con propiedad y en condiciones de seguridad recursos de hardware, software y redes para emplear los ambientes que necesite para el desarrollo y la verificación del software, mantener los repositorios de información que necesite utilizar y disponer de los productos de su trabajo en condiciones de confiabilidad.

Capacidades Transversales

a. Abstracción
Implica descartar o reducir detalles poco significativos de la información sobre un problema para concentrarse en pocos elementos por vez, lo que resulta en una reducción de la complejidad que permita conceptualizar de modo más simple un dominio de problemas para facilitar su comprensión y manejo en forma genérica de sus posibles soluciones.

b. Pensamiento combinatorio
Conduce a la consideración sistemática de un conjunto de alternativas, lo que incluye el manejo mental de muchas variables o detalles del problema sin perder nunca de vista el concepto o la estrategia general de resolución.

c. Autorregulación
Implica manejarse respetando reglas y limitaciones, tanto explícitas como implícitas, sean éstas propias o del grupo de trabajo; actuar ateniéndose a un orden propio que le facilite el acceso a lo que puede necesitar, reconocer y guardar; referenciar la información y registrarla de tal forma que le facilite acceder posteriormente en forma rápida para evaluarla y recuperarla.

d. Comunicarse apropiadamente
Implica una disposición a reconocer que existen otros que pueden aportar información útil o a quienes puede interesarle lo que hace. Supone reconocer su rol y el de cada integrante del proyecto, transmitir la información necesaria en forma precisa y en un lenguaje apropiado para el entendimiento mutuo en interacciones individuales o grupales, o en forma escrita, utilizando, si es necesario para ello, el idioma inglés, que debe interpretar con propiedad a nivel técnico.

e. Trabajar en equipo
Implica adoptar una actitud abierta, estar dispuesto a compartir información y conocimientos, a tomar en cuenta a los usuarios del producto que está construyendo, a brindar, pedir y aceptar ayuda cuando ésta resulte necesaria para facilitar su propia labor o la de otro integrante del equipo. Comprende al equipo del proyecto, incluyendo a los usuarios que participan del mismo.

f. Autoaprendizaje
Implica aprender a capitalizar experiencias a partir de su propio trabajo, a tomar iniciativas para actualizar o profundizar sus conocimientos y habilidades, investigar fuentes de información o herramientas que le pueden resultar útiles. Aplica metodologías de investigación y dedica tiempo a este fin.

Áreas de Actividad

1. Interpretar, en el contexto del proyecto, **especificaciones de diseño** o requisitos de las asignaciones a programar.

	Actividad	**Criterios de realización**
1.1	Comprender, en su contexto inmediato, cuál es el problema a resolver	se interpreta lo especificado observando las reglas de los lenguajes en que está expresado; se puede describir el problema aplicando un mayor nivel de abstracción;
1.2	Determinar el alcance del problema	no se asume lo que no está especificado;
1.3	Validar coherencia e integridad de las especificaciones	se identifican puntos ambiguos o faltantes;
1.4	Convalidar su propia interpretación con quienes la hayan realizado o provisto	se identifican aspectos poco claros o faltantes; se formulan preguntas conducentes a clarificar los requisitos;

2. Planificar su propio trabajo en el contexto del equipo de desarrollo del proyecto.

	Actividad	**Criterios de realización**
2.1	Dividir la asignación en varias menores (por etapas, por productos requeridos, por lo que existe y hay que adaptar o hay que hacer nuevo)	se identifican subtareas o productos intermedios que tienen límites u objetivos claros y resultan medibles (sirven de puntos de control);
2.2	Identificar aspectos críticos (por complejidad, extensión o falta de experiencia) de la asignación	se reconocen dificultades propias de la asignación o en relación a la capacidad propia; se anticipan esfuerzos importantes;
2.3	Estimar tiempos de realización y compararlos con el asignado para la tarea	se estiman lapsos razonables;
2.4	Establecer prioridades y necesidades de apoyo o consulta	se establece un compromiso adecuado entre lapsos y dificultades;
2.5	Participar en sesiones de trabajo en que haya que estimar proyectos	se aportan estimaciones coherentes con las demás;

3. Analizar estrategias para desarrollar la asignación recibida en el contexto del proyecto y de la tecnología a utilizar.

	Actividad	**Criterios de realización**
3.1	Investigar para **refinar aspectos ambiguos** o insuficientemente conocidos del diseño o los requisitos para la solución del problema	se averiguan y completan detalles de diseño no especificados en la asignación recibida; se resuelven problemas de lógica que implican diseño o adaptación de algoritmos o utilización de estructuras de datos que faciliten o permitan la solución;
3.2	En el caso de asignaciones cuyo objetivo es mantener programas (realizar cambios) o resolver errores observados, **encontrar la causa** que motiva el error **o el lugar** en que hay que introducir el cambio.	se recorren y descartan sistemáticamente distintas posibilidades; si existe más de un lugar a corregir, se encuentran todos antes de realizar correcciones;
3.3	Identificar fuentes confiables de información	se relacionan tecnologías o tipo de componentes

	Utilizar metodologías de búsqueda y selección de información	con sitios que contienen material; se consideran pros y contras;
3.4	Detectar componentes reusables como insumo	se identifican patrones, clases, objetos, rutinas o subprogramas que estén disponible en bibliotecas u otras fuentes de software y puedan ser utilizados como componentes de la solución requerida;
3.5	Construir prototipos (pruebas de concepto) y demos para ser utilizados para analizar la propiedad de soluciones	los prototipos o demos dan una imagen representativa de lo que se propone; el programador introduce con agilidad las propuestas de los usuarios;
3.6	Evaluar distintas estrategias de solución	se eligen soluciones eficaces o adecuadas de acuerdo a criterios consistentes con los objetivos del proyecto (eficiencia, claridad, rapidez);

4. Producir el código que resolverá la asignación en el contexto de la tecnología a utilizar. Esto incluye la modificación (agregado, reemplazo o eliminación) de código ya escrito, tanto sea para corregir errores observados en pruebas o para cambiar funcionalidades o el comportamiento de productos con existencia previa.

	Actividad	**Criterios de realización**
4.1	Desarrollar algoritmos que den solución a los problemas asignados o los derivados de los mismos	Los pruebas unitarias de los algoritmos dan los resultados previstos y los algoritmos son seguros, eficientes y fáciles de adaptar.
4.2	Definir estructuras de datos eficaces para manejar los datos requeridos y explotarlas con eficiencia	Se crean estructuras temporarias canónicamente correctas El acceso a los datos resulta eficiente bajo distintas condiciones de uso previsibles
4.3	Definir, instanciar y completar clases y objetos apropiados para representar los problemas a resolver	Se respetan principios de “data hiding” Se utilizan conceptos de herencia y polimorfismo
4.4	Armar interfaces apropiadas al problema y al usuario, respetando el estilo del contexto previsto	Se aplican criterios de simplicidad y coherencia Se toman en cuenta posibles errores y se brinda información útil sobre los mismos
4.5	Identificar y guardar clases, objetos, rutinas o subprogramas (o sus referencias) que resulten interesantes para ser utilizados como componente en alguna otra oportunidad	Construye o aporta a una biblioteca de clases y objetos debidamente documentados y en la que resulta fácil encontrar lo que se busque.

5. Verificar los productos desarrollados.

	Actividad	**Criterios de realización**
5.1	Analizar y registrar todos los procesos alternativos que contiene el producto elaborado	No se pasan por alto alternativas importantes
5.2	Desarrollar un juego de prueba conceptual que cubra esas alternativas	El juego de prueba cubre razonablemente las alternativas posibles Se toma en cuenta el contexto del sistema en el cual se insertará su asignación
5.3	Implementar el juego de prueba utilizando un ambiente de software apropiado	El juego de prueba es una instancia válida del punto anterior
5.4	Procesar la prueba, obteniendo y registrando los resultados o anormalidades que se observen	Se trabaja metódicamente Se registran apropiadamente los incidentes encon-

		trados y toda la evidencia pertinente a los mismos
5.5	Buscar las causas de los incidentes encontrados	Se trabaja metódicamente recorriendo sistemáticamente el código y dejando registros para contrastar resultados parciales

6. Depurar código de programas.

	Actividad	**Criterios de realización**
6.1	Relacionar resultados insatisfactorios de pruebas con probables causas y rastrearlas hasta los datos o porciones de código que los originaron	No quedan incidentes sin que se encuentre la causa que los originó. Se reconocen los errores propios tanto como los ajenos.
6.2	Relacionar situaciones informadas en partes de incidentes con probables causas y rastrearlas hasta los datos o porciones de código que los originaron	Se identifican las porciones de código que originaron el malfuncionamiento.
6.3	Analizar los datos y/o partes del código que originaron el mal funcionamiento y determinar conceptualmente el tipo de corrección o reemplazo	Se identifica el error en el código que produjo el malfuncionamiento. No quedan sin descubrir otras porciones de código que también puedan ocasionar el mismo tipo de malfuncionamiento.
6.4	Analizar el posible reemplazo y evaluar la posibilidad de que la corrección o reemplazo introduzca otros problemas	El reemplazo soluciona el malfuncionamiento. El reemplazo no introduce nuevos problemas.
6.5	Evaluar las ineficiencias reportadas en la utilización de recursos (tiempo, memoria, otros) que hace el producto	Si corresponde, se replantea el enfoque de la solución

7. Realizar, con otros programadores o con especialistas, **revisiones cruzadas de código o de interfaces**.

	Actividad	**Criterios de realización**
7.1	Revisar el cumplimiento de estándares, de buenas prácticas y de especificaciones	El código original o las propuestas de cambio responden a normas de buena práctica Resultan piezas de código mantenibles y bien documentadas
7.2	Revisar las interfaces con especialistas (CHI)	Las interfaces originales o las proposiciones de cambio responden a buenas prácticas de diseño de interfaces hombre-máquina Resultan interfaces coherentes dentro del estilo del sistema, amigables para el usuario y que no ofrecen problemas a personas con capacidades diferentes
7.3	Evaluar el uso eficiente y completo de recursos y del ambiente	Resultan piezas de código que no demanden tiempos de proceso o asignaciones de memoria o almacenamiento excesivos para el contexto Si corresponde, se recomiendan cambios en el enfoque de la solución
7.4	Reportar observaciones sobre propuestas de cambio	Se informan observaciones o se presentan propuestas de cambio que son significativas Emplea tacto y diplomacia para presentar sus conclusiones en forma verbal o escrita

8. Documentar su trabajo comentando en línea el código y las clases.

	Actividad	**Criterios de realización**
8.1	Describir características, relaciones y limitaciones de nuevas clases que se introduzcan utilizando diagramas u otros elementos	Claridad, consistencia y completitud de la documentación resultante
8.2	Intercalar en el código del producto descripciones de sus características y limitaciones, así como también decisiones de diseño que puedan ayudar a otros a interpretar lo hecho	Un lector del código entiende qué hace cada parte del código y por qué se incluyó.
8.3	Registrar elementos utilizados y resultados de pruebas	Están disponibles los datos, otros elementos o la situación que lo originaron
8.4	Registrar incidentes, errores, soluciones y tiempos empleados para documentar el proceso de desarrollo	Existen registros y evidencias de las actividades realizadas y de los incidentes observados
8.5	Identificar las diferentes versiones del producto	Cada versión está identificada de acuerdo a estándares (ver 9-e)

Desempeño de base

9. Explotar las funcionalidades de los sistemas informáticos, hardware, software y redes para la realización de sus actividades.

	Actividad	**Criterios de realización**
9.1	Configurar lógicamente el sistema al entorno de trabajo para desarrollar y probar programas	No se generan conflictos entre componentes de software, hardware o usuarios.
9.2	Organizar y mantener componentes de software y datos de prueba en sistemas de archivos, utilizando las utilidades comunes al proyecto	Se accede con facilidad a lo que se requiere. No se extravían ni se confunden archivos requeridos
9.3	Recuperar, presentar y distribuir información (diagramas, pantallas, etc.) por dispositivos de salida y comunicaciones disponibles en su estación de trabajo o a través de la red	La documentación integra efectivamente ejemplos, diagramas y otros elementos que facilitan su comprensión. La documentación resulta accesible por quien la necesita.
9.4	Respetar procedimientos propios o de la organización que aseguren la integridad, disponibilidad y seguridad del sistema y de la información durante el desarrollo y verificación de programas	Se respetan estándares de seguridad, aún en las pruebas de sistemas. Se identifican y preservan versiones de programas.
9.5	Integrar la producción propia en el conjunto del proyecto identificándolas de acuerdo a los procedimientos de administración de versiones en uso por el proyecto	Cada versión está identificada de acuerdo a estándares El Administrador de Configuraciones no encuentra problemas en su tarea

Campo y condiciones del ejercicio profesional

Principales resultados del trabajo
Unidades (módulos, objetos, componentes, programas) que cumplen con la asignación recibida -especificación de diseño-, responden a buenas prácticas y normas de programación del proyecto o la organización para los que fueron elaborados -lo que significa que sean entendibles y mantenibles por otros-, y no contienen fallas o malfuncionamientos intrínsecos dentro de la tecnología o ambiente para los cuales fueron desarrollados.
Reportes de actividad o de incidentes, así como sugerencias de cambios a partir de revisiones cruzadas de código.
Identificación y catalogación de unidades de programación reutilizables.

Medios de producción o de tratamiento de la información
Entornos de programación compuestos por equipamiento y software de base, ambientes de programación compuestos por editores de texto y de diagramas, compiladores, generadores o intérpretes, catálogos de patrones, bibliotecas de programas y componentes, otras herramientas de software para preparar y realizar pruebas, diagnóstico, recuperación de datos.
Internet, correo electrónico, foros y listas de discusión.

Procesos, técnicas y regulaciones normativas
Lenguajes de programación y ambientes de desarrollo de programas.
Manuales de estilo y buenas prácticas.
Normas ISO, CCITT e IEEE. Normas aplicables a las comunicaciones digitales.

Datos e información disponibles o generados
Utiliza especificaciones de diseño de software y otra documentación del proyecto.
Genera informes sobre trabajo realizado y reportes de incidentes.

Espacio social de trabajo: relaciones funcionales o jerárquicas
Trabaja en relación formal o informal de dependencia con el proyecto en el que se desempeña, integrando y colaborando con el equipo de trabajo del mismo. Es supervisado jerárquica y técnicamente por el líder del proyecto o del grupo, de quien recibe las asignaciones de trabajo y a quien solicita consejo y asesoramiento, consensuando enfoques o cronogramas de actividad.
Intercambia información, recibe o brinda asesoramiento a sus pares o a otros especialistas, participa en reuniones de su equipo y en revisiones cruzadas de su trabajo o el de otros.
En la mayoría de los casos no tiene personal a cargo.

Proceso de Consulta

Análisis Ocupacional

Recomendaciones del Plan Estratégico 2004-2014 del Foro de la Competitividad para el Sector de Software y Servicios Informáticos organizado por la Secretaría de Industria, con participación de la CEPYME, Ministerios de Educación, Ciencia y Tecnología, de Trabajo, Empleo y Desarrollo Social y de Relaciones Exteriores, como también de representantes de entidades, organizaciones, empresas y universidades.

Entrevistas a Líderes de Proyecto, Coordinadores de Programación y Psicólogos encargados de Selección de Personal de las siguientes organizaciones:

- Data-Waves S.A.
- G&L Group
- Siemens Itron
- Accenture
- Novamens
- Hexacta S.A.
- Serbal (Cluster Córdoba Technology)

Reuniones con la Comisión de RR.HH. y Capacitación de CESSI, Cámara de Empresas de Tecnología, con representantes de la Comisión Directiva de CICOMRA, Cámara de Informática y Comunicaciones de la República Argentina y con socios titulares y representantes del Cluster Córdoba Technology.

Talleres de Reflexión sobre las actividades realizadas por los programadores realizados en:

Ciudad de Buenos Aires, 8 y 9 de febrero de 2006

Ciudad de Córdoba, 17 y 18 de abril de 2006

con la participación de programadores, analistas programadores, supervisores y responsables de investigación y desarrollo seleccionados por su desempeño profesional por empresas del medio local a partir de invitaciones cursadas a las asociaciones de primer grado.

Validación del Perfil Profesional

El Perfil fue revisado y validado por un Grupo de Trabajo compuesto por representantes de CESSI, Cámara de Empresas de Tecnología, del Cluster Córdoba Technology, de CICOMRA, Cámara de Informática y Comunicaciones de la República Argentina, del Polo IT de Buenos Aires y del Polo IT de Rosario, en reuniones mantenidas durante el mes de junio.

También fue presentado ante la Comisión de RR.HH. y Capacitación de la CESSI, Cámara de Empresas de Tecnología.

También fue revisado en una reunión del Foro de la Competitividad para el Sector de Software y Servicios Informáticos organizado por la Secretaría de Industria el 22 de agosto de 2006.