Universidad Autónoma de Baja California

Facultad de ingeniería, Arquitectura y diseño

Ingeniería en Software y Tecnologías Emergentes

Paradigmas de la Programación

Docente: Gallegos Mariscal José Carlos

Alumno: Diarte Salas Gilberto

Matricula: 360954

Grupo: 941

Práctica: Paradigma funcional, Tour Standard ML

Fecha de Entrega: 31/mayo/2024

Introducción

Standard ML (SML) es un lenguaje de programación funcional de propósito general, conocido por su tipado estático y potente inferencia de tipos. Esto significa que, aunque los errores de programación comunes son prevenidos por el sistema de tipos, la necesidad de declaraciones explícitas de tipo es mínima. Esto facilita la creación de software que es fácil de entender, extender y mantener. Existen compiladores optimizadores gratuitos para SML, como MLton, que generan código nativo eficiente. Además, la extensión de concurrencia "Concurrent ML" permite el soporte de procesos secuenciales que se comunican, siendo compatible con SML/NJ y MLton.

Desarrollo

El lenguaje Standard ML tiene varias características y estructuras que facilitan su uso y potencian sus capacidades. Algunas de las características clave incluyen:

Valores y Tipos Básicos:

En SML, se pueden definir valores usando la palabra clave val. Aunque se pueden hacer declaraciones explícitas de tipo, generalmente no son necesarias debido a la inferencia de tipos.

Los valores son inmutables, aunque se pueden redefinir nombres para darles nuevos valores.

Estructuras de Control y Funciones:

SML permite la creación de funciones usando fn y fun, y todas las funciones en SML toman un solo argumento y están curried, es decir, se pueden aplicar parcialmente.

Las expresiones let permiten hacer declaraciones con alcance local.

Módulos y Firmas:

Los programas en SML se componen de módulos, que agrupan un conjunto de definiciones. Un módulo puede exponer todas sus definiciones por defecto, pero esto se puede controlar mediante firmas (signatures), que definen qué partes del módulo son visibles externamente.

Los funtores (functors) permiten que los módulos acepten otros módulos como parámetros, facilitando la modularidad y reutilización de código.

Definición de Tipos y Tipos Recursivos:

Se pueden definir alias de tipos con la palabra clave type y nuevos tipos de datos con datatype.

Los tipos recursivos, como listas y árboles, se definen fácilmente, permitiendo estructuras de datos complejas y recursivas.

Patrones y Coincidencia de Patrones:

SML permite descomponer valores y evaluar diferentes casos usando coincidencia de patrones, lo que facilita la manipulación y análisis de datos.

La coincidencia de patrones también proporciona verificación de exhaustividad, alertando al programador si falta alguna rama en una estructura de coincidencia.

Concurrencia con Concurrent ML:

Concurrent ML extiende SML para soportar concurrencia, proporcionando hilos ligeros (spawn), canales de comunicación (send, recv), y variables compartidas seguras (IVars, MVars).

Las operaciones de comunicación son eventos, y se puede esperar múltiples eventos usando select.

Conclusión

Standard ML es un lenguaje poderoso y versátil que combina las ventajas de la programación funcional con un sistema de tipos fuerte y flexible. Su capacidad para manejar concurrencia de manera eficiente mediante Concurrent ML lo hace adecuado para aplicaciones modernas que requieren procesamiento paralelo. La modularidad de SML, junto con su enfoque en la inmutabilidad y la coincidencia de patrones, promueve la creación de código claro, mantenible y seguro. Para aquellos interesados en la programación funcional y la teoría de tipos, SML ofrece una experiencia robusta y enriquecedora.