Diseño de Modelos de Programación Distribuida

Los modelos de programación distribuida que Meiklejohn reconoce son:
-ARGUS
-Programming Language in the Sky (PLITS)
-Remote Procedure Call
-Hermes
-Emerald
-Promises

Cualquier modelo de programación distribuida debe estar dentro de los límites especificados por el teorema de CAP. Entonces, en un extremo están los modelos AP y en el otro los modelos CP.
Lasp es un ejemplo de un modelo AP, sacrifica la consistencia por la disponibilidad, las estructuras de datos son CRDT y, por lo tanto, todas las operaciones conmutan y las estructuras de datos se expresan como cerradas-semicubiertas. Hay restricciones sobre los tipos de estructuras de datos que soporta Lasp.
Spry es un modelo de programación para la creación de aplicaciones que intercambien disponibilidad y consistencia en diferentes partes del código para cumplir con los requisitos de la aplicación.

FLP establece que, en un sistema asíncrono, el acuerdo es imposible cuando hay un solo proceso fallido, no podemos distinguir entre una respuesta tardía y una falla.
El teorema de CAP establece que cuando algunos procesos de un sistema no se pueden comunicar entre sí (están particionados), tenemos que sacrificar la disponibilidad si queremos mantener la linealidad (consistencia).

Existen 2 resultados imposibles que demuestran porque estos problemas son difíciles y establecen los límites de lo que es posible: el teorema de CAP y el resultado de FLP.

La concurrencia tiene como principal problema que los procesos compiten por los recursos del sistema por lo cual el acceso debe der controlado, el SO debe ofrecer mecanismos para sincronizar la ejecución de procesos

Los modelos de programación usados hoy en día son del estilo del modelo de von Neumann, donde la computación usa ubicación de almacenamiento y el avance de los programas está en un solo hilo de ejecución.
La programación concurrente extiende esto para adaptar múltiples hilos secuenciales de ejecución

La programación distribuida es una extensión de la concurrencia, así como ésta se extiende secuencialmente.
La programación distribuida es diferente de la programación concurrente, las maquinas que se comunican en una red a veces son completamente fallidas, incapaces de responder a una solicitud, etc.
Por lo que las herramientas que existen son insuficientes para resolver los problemas de programación distribuida y nos referimos a esta clase de fallas como “falla parcial”

El problema principal en los sistemas distribuidos es el del acuerdo o consensuado y se presenta de 2 formas, la Elección del líder, donde el proceso debe establecer un líder activo en un grupo de nodos, y la Detección de fallas, donde el proceso debe detectar cuando un nodo falla y evitar que sea líder