Dominio determinado por el diseño, Eric Evans:

1. Enlazar el modelo y la aplicación. Es una vinculación esencial que se mantendrá durante todo el proceso.

2. Cultivar un lenguaje basado en el modelo. Esto ayuda a la organización y al entendimiento.  
3. El desarrollo de un modelo rico en conocimientos. El modelo es más que un esquema de datos, ya que se integra a la solución de un problema complejo.

4. Destilación del modelo. Agregar conceptos importantes al modelo para hacerlo más completo y también eliminar conceptos que no resulten útiles.   
5. Lluvia de ideas y la experimentación.    
Es la creatividad de intercambio de ideas y la experimentación masiva, que permiten  
encontrar un modelo rico en conocimientos y destilarlo.

Conocimiento crujido:

No es una actividad solitaria. Un equipo de desarrolladores y expertos del dominio colaboran, por lo general dirigidos por los desarrolladores. La materia prima proviene de la mente de los expertos del dominio. En el método de cascada de edad por ejemplo, los expertos de negocios hablan con los analistas,   
y los analistas piensan el problema en abstracto y lo pasa los programadores. Este enfoque produce un error porque  carece por completo de retroalimentación. Los analistas se encargan exclusivamente de crear el modelo, basándose únicamente en las aportaciones de los expertos de negocios.   
Ellos no tienen oportunidad de aprender de los programadores o ganar experiencia con las primeras versiones de software. El conocimiento se escurre en una dirección, pero no se acumula.

Los buenos programadores, naturalmente, comienzan a desarrollar un resumen y modelo del trabajo. Pero cuando esto ocurre   
sin la colaboración de expertos de dominio, los conceptos   
son ingenuos. Esa superficialidad del conocimiento produce software que hace un trabajo de base, pero carece de una conexión profunda con la forma de pensar del experto de dominio. La interacciones entre los miembros del equipo son muy importantes. El perfeccionamiento constante de los desarrolladores y de los expertos de dominio ayuda a una mayor comprensión entre los miembros del equipo y también hacia lo que se va a desarrollar.   
Surgen etapas de aprendizaje continuo, nunca sabemos lo suficiente. El conocimiento  
en el proyecto se encuentra fragmentado, disperso entre muchas personas y documentos. Hay caminos que parecen  
mejores pueden engañar. No se es consciente de lo mucho que no se sabe y esta ignorancia nos lleva a hacer falsas suposiciones.  
Puede ocurrir que que el proyecto original se deje de lado ya que los expertos del dominio comprendieron mas y aclararon lo que realmente buscaban, ahun asi los elementos claves del modelo nos pueden servir. Tambien algo muy importante es el conocimiento adquiridos por los miembros del equipo, desarrolladores y expertos en el dominio.

Modelos Profundos  
los modelos útiles rara vez se encuentran en la superficie. Cuando hemos llegado a comprender el dominio y las necesidades de la aplicación, tenemos que desechar elementos superficiales del modelo que nos parecieron importantes en el principio, o tenemos que cambiar su perspectiva.

Principios de diseño de Smalltalk, de Daniel H. H.Ingalls

principios generals del sistema smalltalk:

**Dominio Personal:***Si un sistema quiere para el espíritu creativo, debe ser completamente entendible para un individuo solitario.* Ingalls *sostiene que c*ualquier barrera que exista entre el usuario y alguna parte del sistema será finalmente una barrera a la expresión creativa.

**Buen Diseño:** *Un sistema debería ser construido con un mínimo conjunto de partes no modificables; esas partes debieran ser tan generales como sea posible; y todas las partes del sistema deberían estar mantenidas en un esquema uniforme.*

**Lenguaje**

Al diseñar un lenguaje Ingalls sostiene que los modelos de computación deben ser compatibles con la mente, en vez de hacerlo al revés. La comunicación entre dos personas (o entre una persona y una computadora) incluye comunicación en dos niveles. La comunicación explícita incluye la información que es transmitida en un determinado mensaje. La comunicación implícita incluye las suposiciones relevantes comunes a los dos seres.

Propósito del lenguaje: *Proveer un esquema para la comunicación.*

Alcance: *El diseño de un lenguaje para usar computadoras debe tratar con modelos internos, medios externos, y con la interacción entre ellos tanto en el humano como en la computadora.*

### Objetos que se Comunican

### Objetos: *Un lenguaje de computación debe soportar el concepto de "objeto" y proveer una manera uniforme de referirse a los objetos del universo.*

### Administración del Almacenamiento: *Para ser auténticamente "Orientado a Objetos" un sistema debe proveer administración automática del almacenamiento.*

### Mensajes: *La computación debería ser vista como una capacidad intrínseca de los objetos que pueden ser invocados uniformemente enviándoles mensajes.*

### Metáfora Uniforme: *Un lenguaje debería ser diseñado alrededor de una metáfora poderosa que pueda ser aplicada uniformemente en todas las áreas.*

### Organización

### Modularidad: *Ningún componente en un sistema complejo debería depender de los detalles internos de ningún otro componente.*

### Clasificación: *Un lenguaje debe proveer un medio para clasificar objetos similares, y para agregar nuevas clases de objetos en pie de igualdad con las clases centrales del sistema.*

### Polimorfismo: *Un programa sólo debería especificar el comportamiento esperado de los objetos, no su representación.*

### Factorización: *Cada componente independiente de un sistema sólo debería aparecer en un sólo lugar.*

### Reaprovechamiento: *Cuando un sistema está bien factorizado, un gran reaprovechamiento está disponible tanto para los usuarios como para los implementadores.*

### Máquina Virtual: *una máquina virtual establece un marco para la aplicación de tecnología.*

### Interfaz al Usuario

### Principio Reactivo: *Cada componente accesible al usuario debería presentarse de una manera entendible para ser observado y manipulado.*

### Sistema Operativo: *Un sistema operativo es una colección de cosas que no encajan dentro de un lenguaje. No debería existir.*

### Trabajo futuro

### Selección Natural: *Los lenguajes y sistemas que son de buen diseño persistirán, sólo para ser reemplazados por otros mejores.*