

Обобщенное и генеративное программирование

Выделение доменов в приложении

Домен задачи:

концепты и
свойства в
терминах задачи

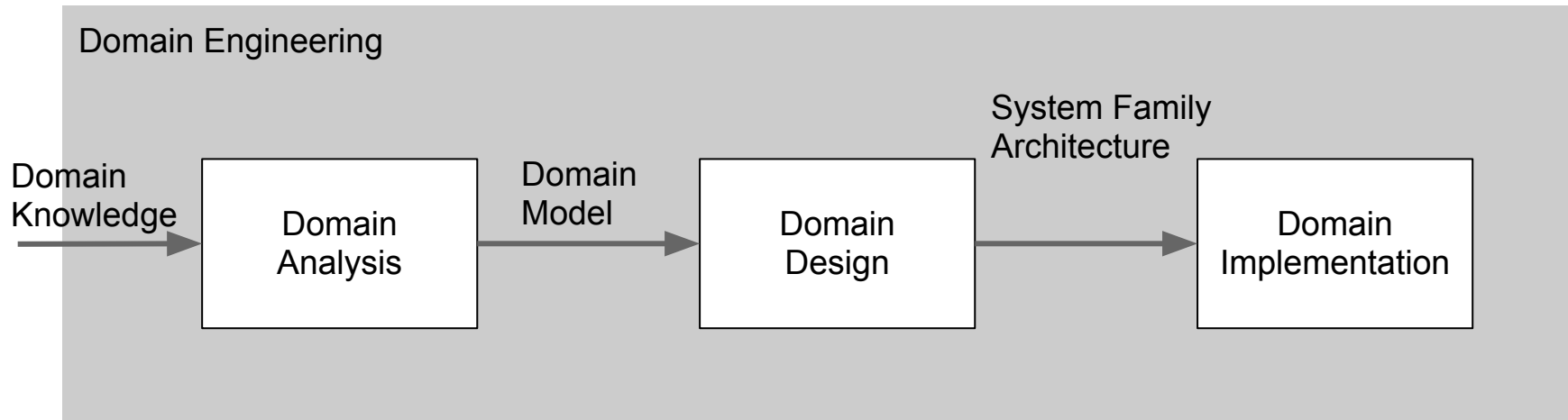
Конфигурация:

- ограничения на свойства
- зависимости
- умолчания
- правила посторения
- оптимизация

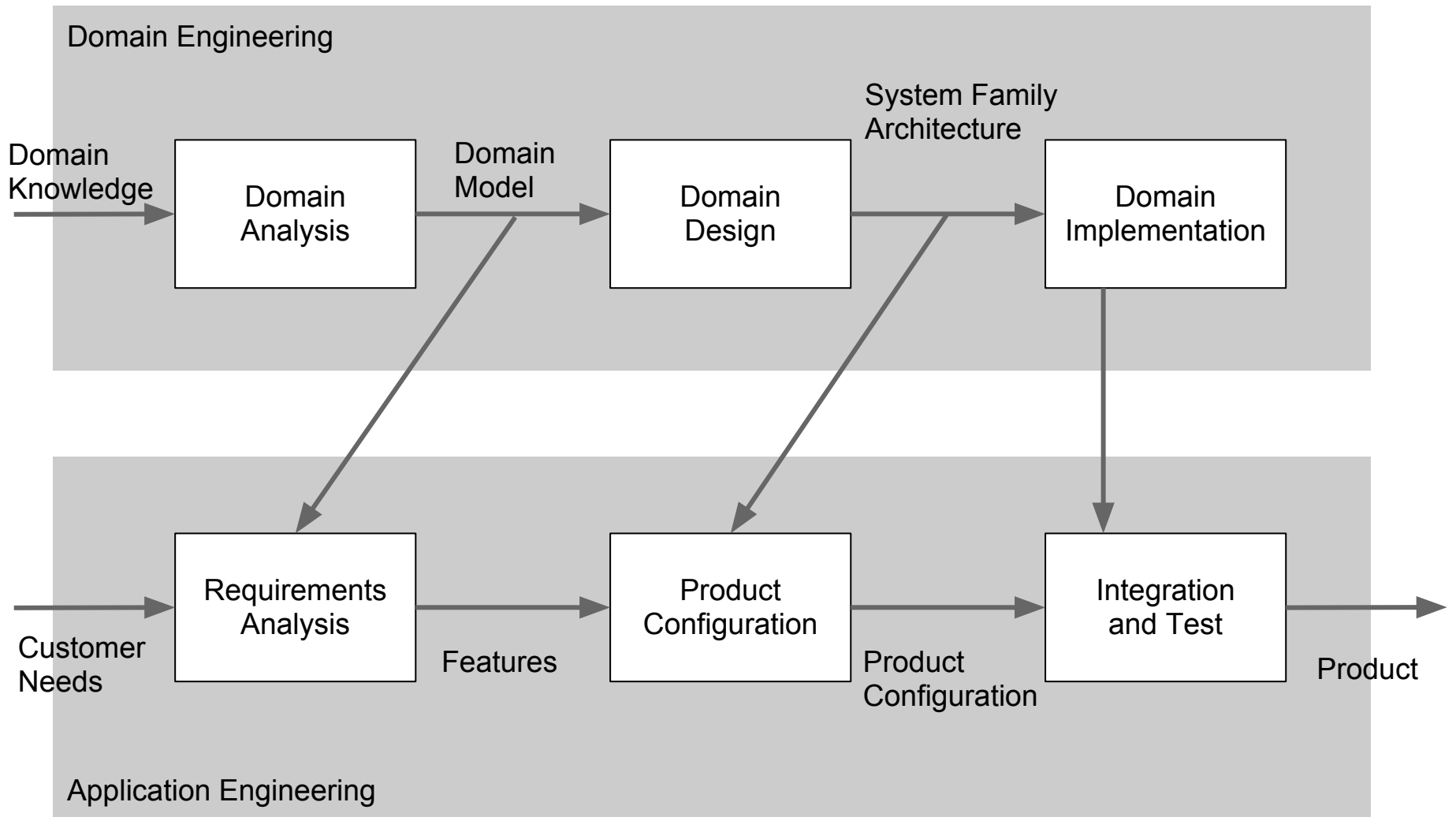
Домен решения:

- компоненты
- избыточность
- совместимость

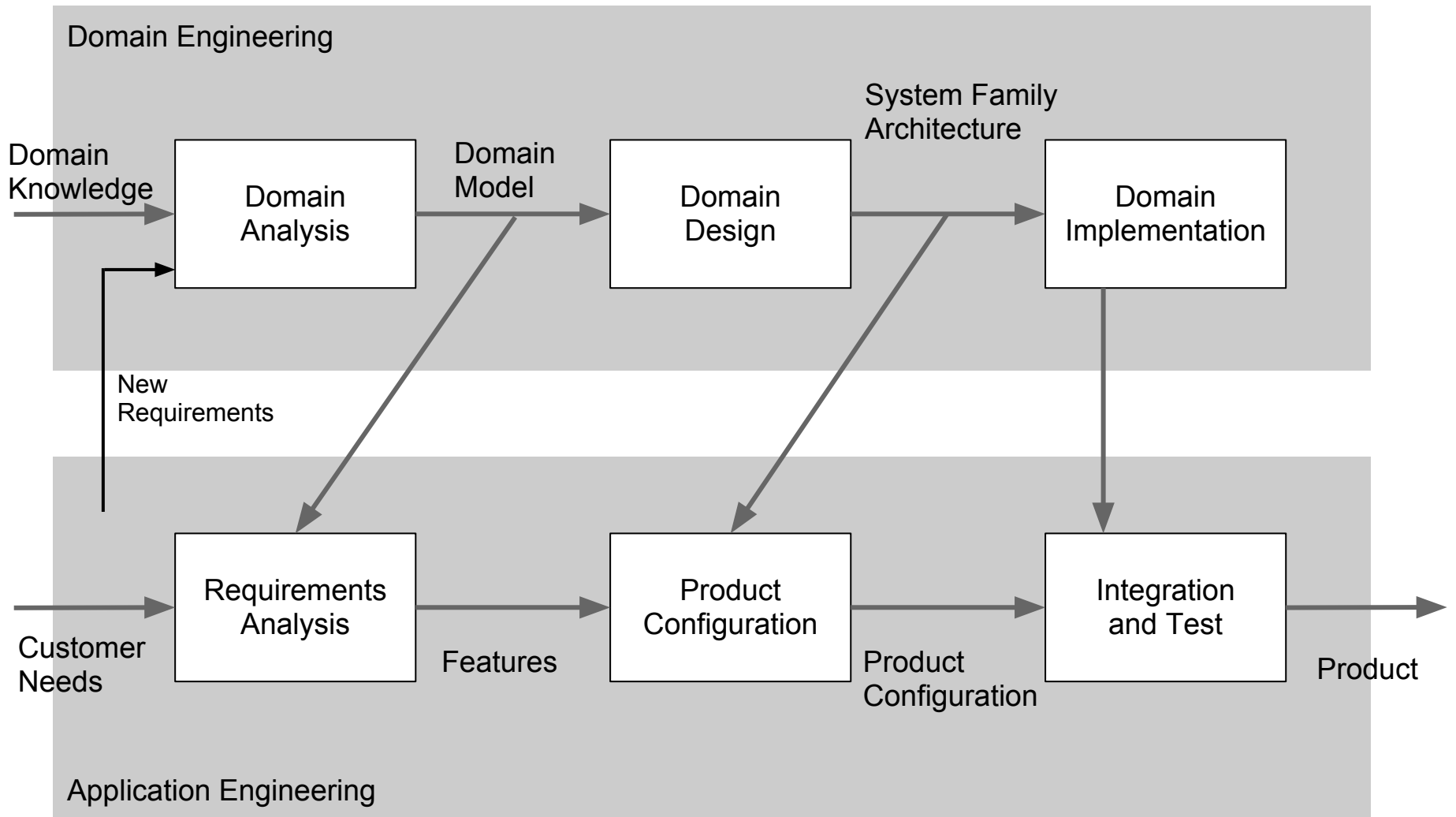
Domain Engineering



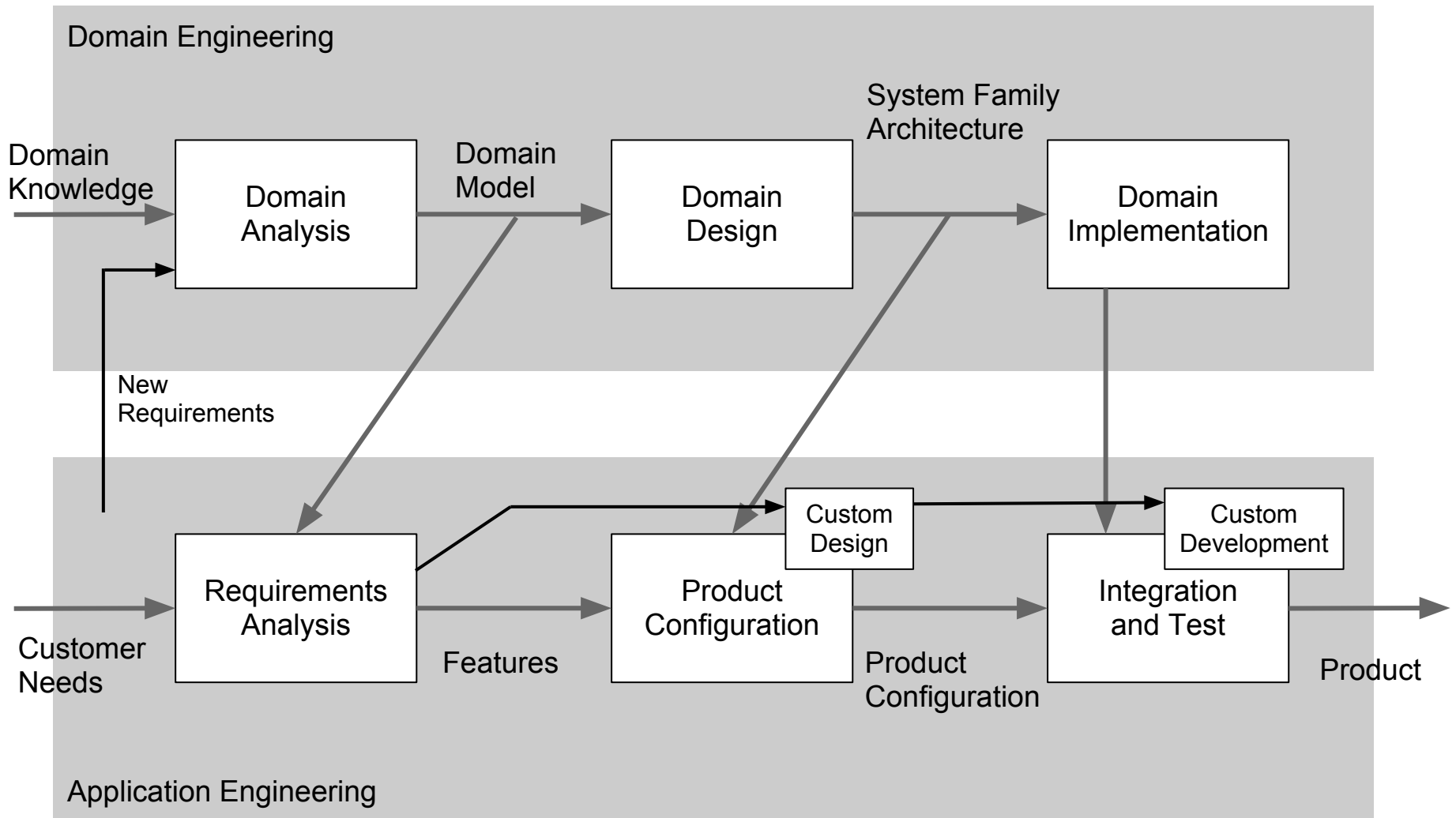
Application Engineering



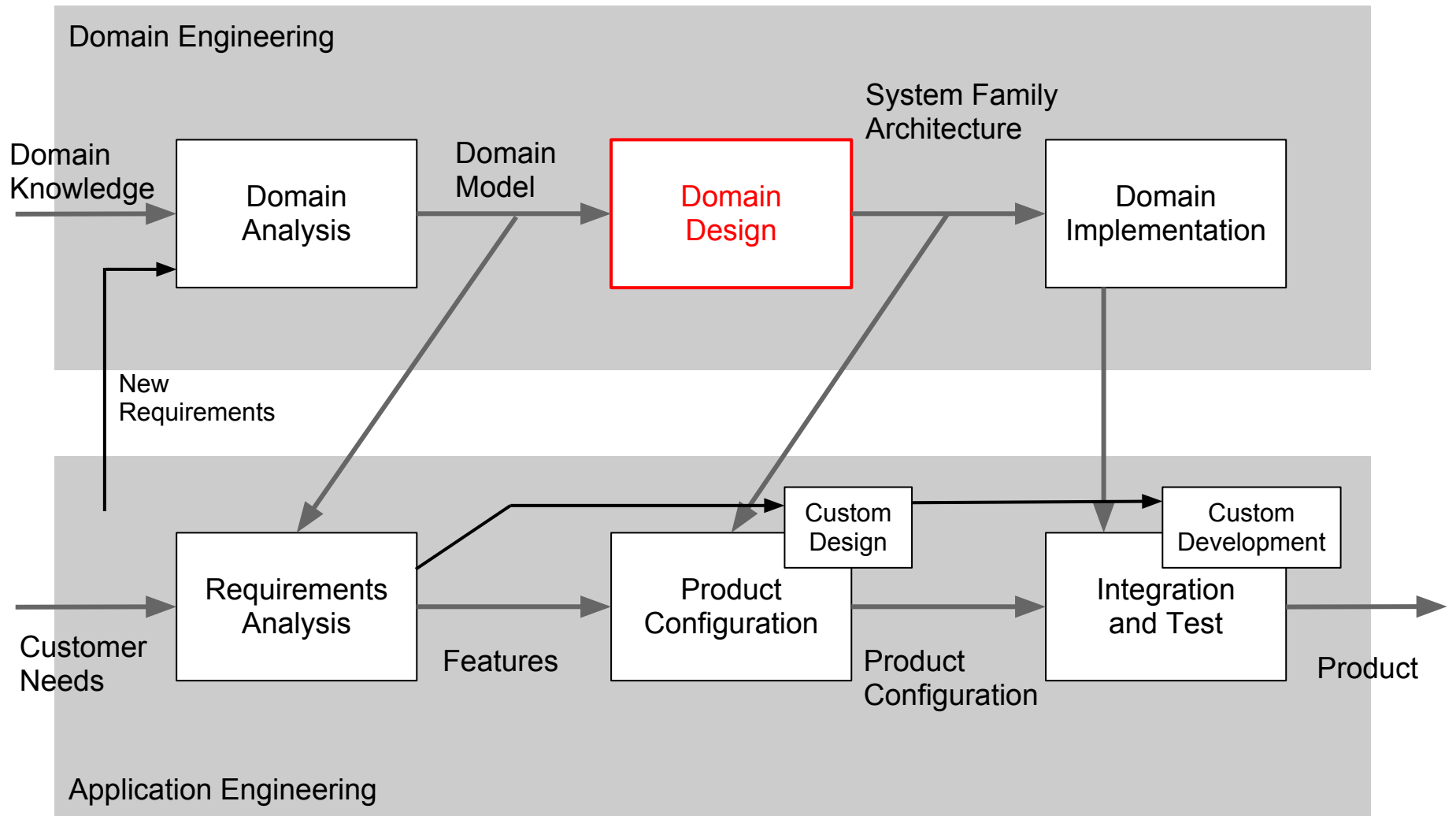
Application Engineering



Application Engineering



Domain Engineering

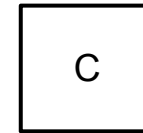


Feature Modeling

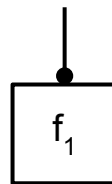
- FODA (Feature Oriented Domain Analysis)
- Concept
- Feature, Subfeature
 - количественные
 - сущностные
- Feature Diagram
 - features
 - constraints
 - semantic description

Элементы диаграммы

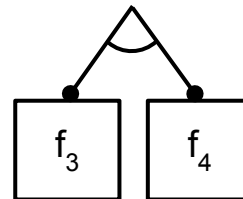
Concept - корневой элемент



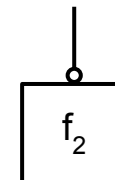
Feature:
mandatory



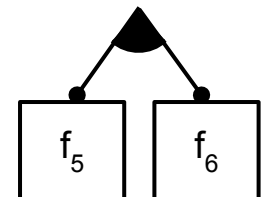
alternative



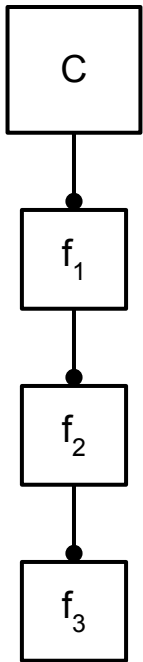
optional



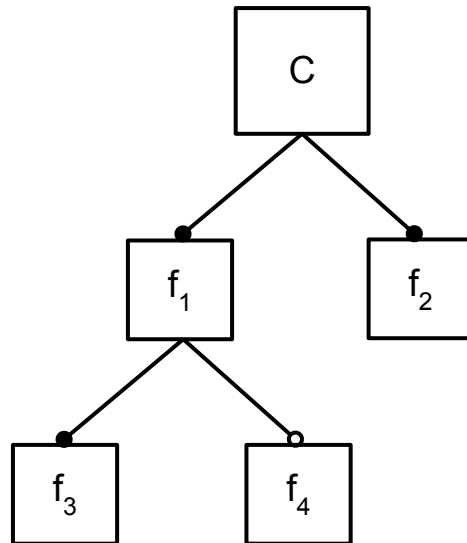
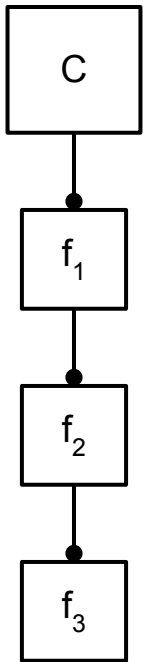
or-feature



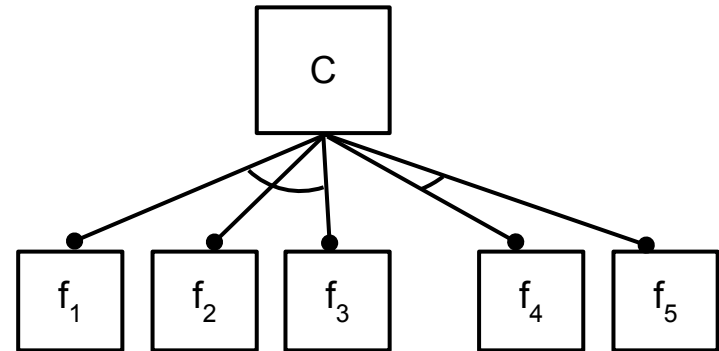
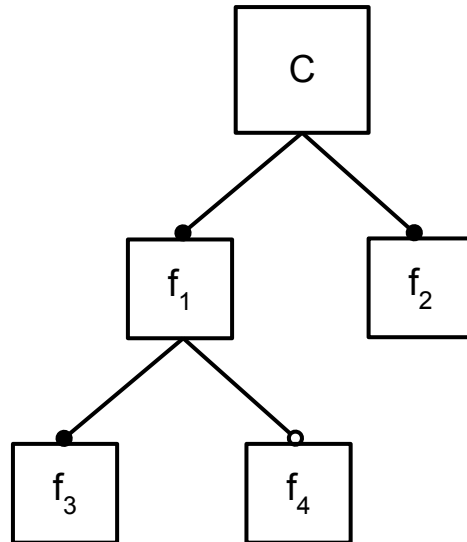
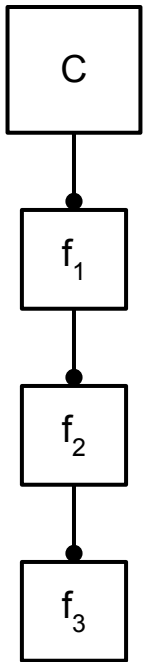
Примеры диаграмм



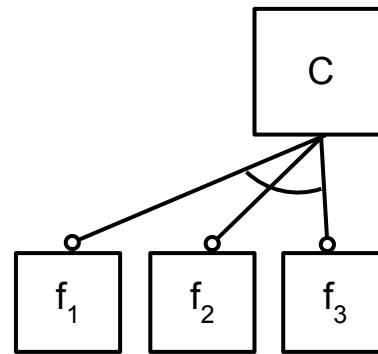
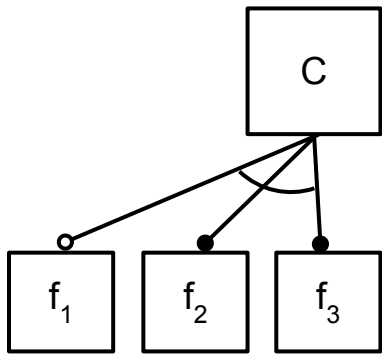
Примеры диаграмм



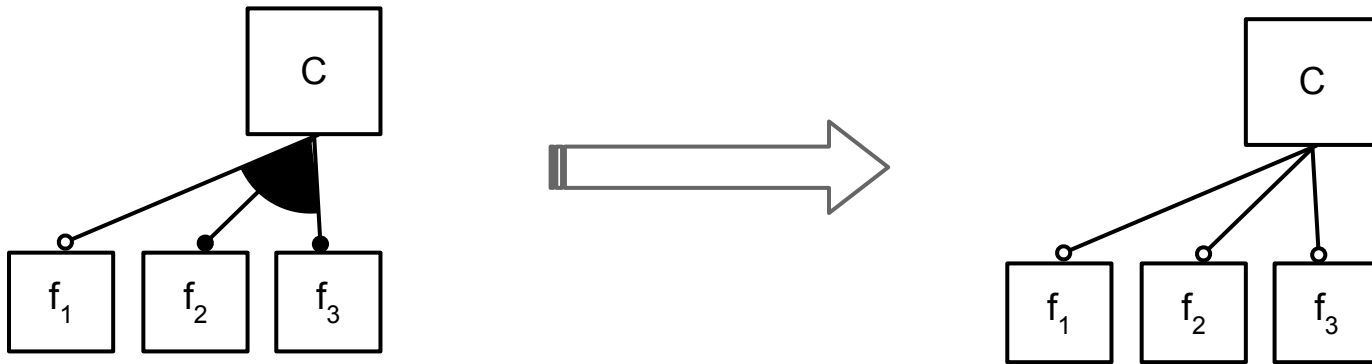
Примеры диаграмм



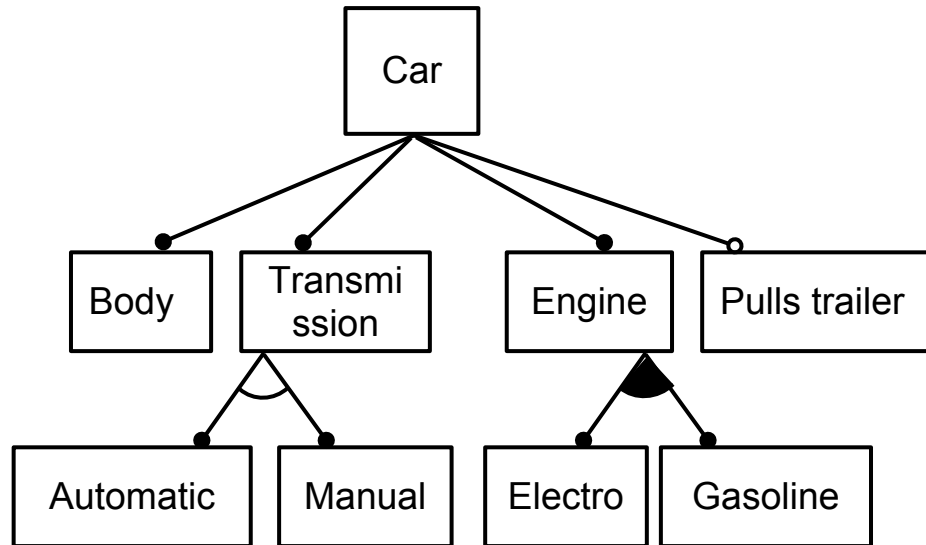
Нормализация диаграмм



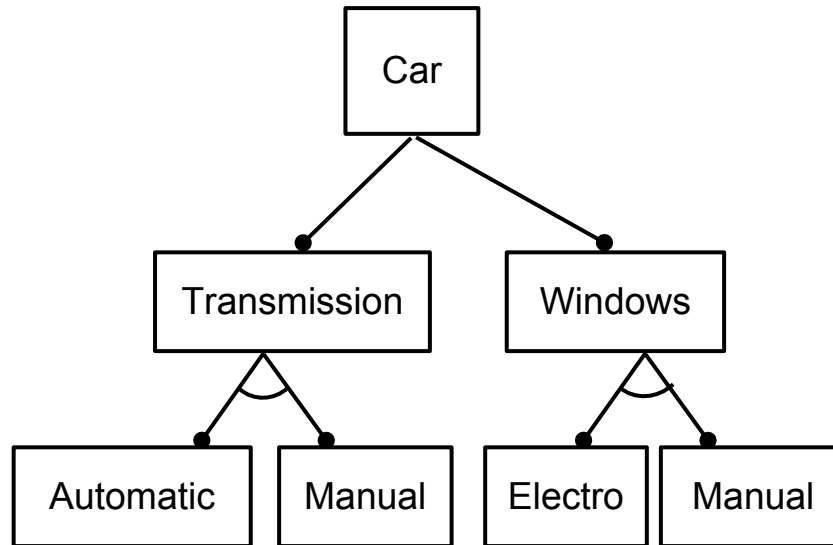
Нормализация диаграмм



Пример моделирования



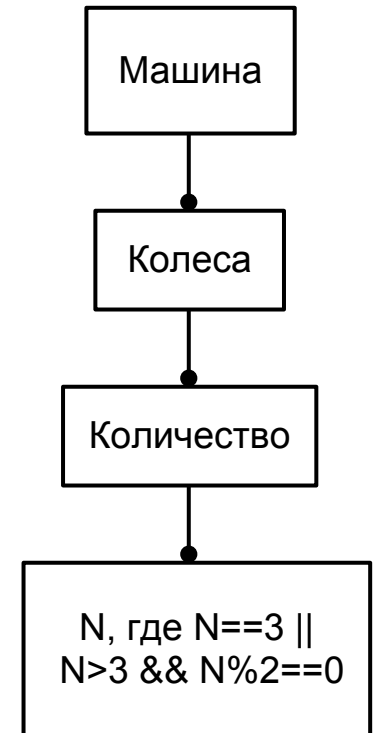
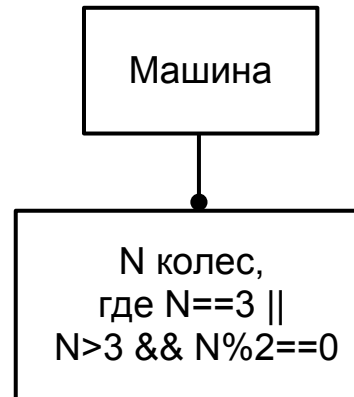
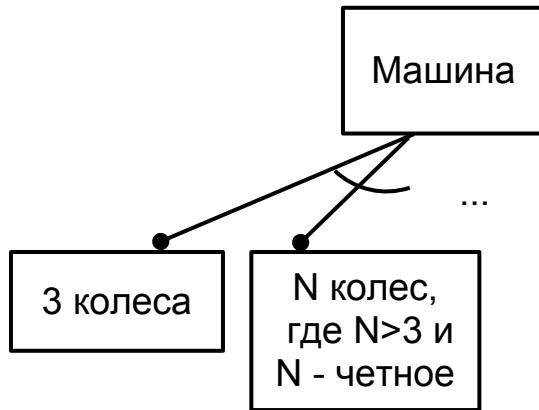
Пример моделирования



Представление модели

Разные системы: UML, XML, DSL

Многообразие представлений



MPS - structure language

concept

concept extension

abstract concept

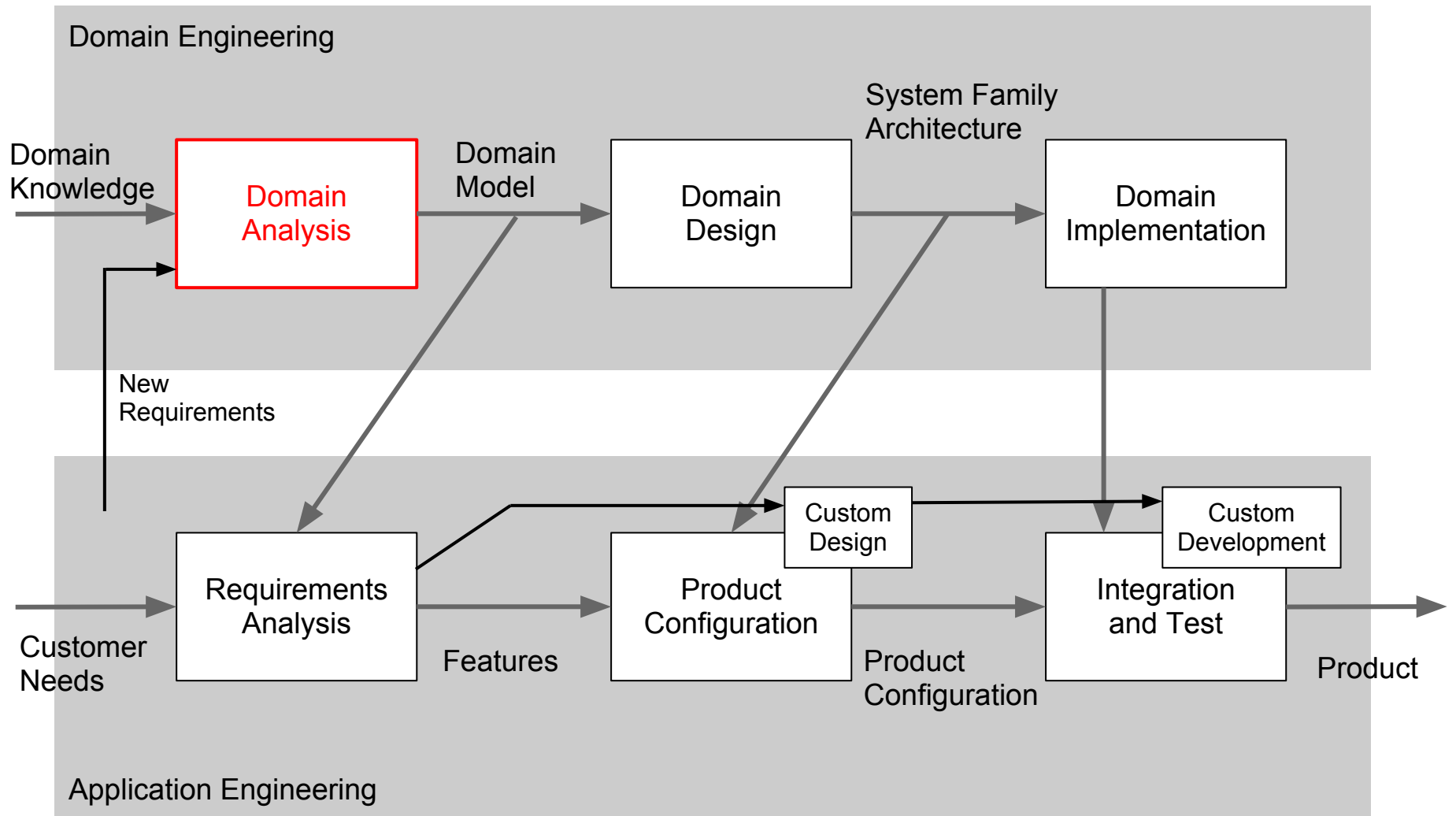
children

properties

Редактирование модели

- Графические редакторы (UML)
- Текстовые редакторы
 - XML
 - редактор + парсер DSL
- Проекционный редактор MPS
 - Model-View-Controller
 - Ячейка (editor cell)
 - Преобразование модели по вводу
 - Набор стандартных ячеек

Domain Engineering

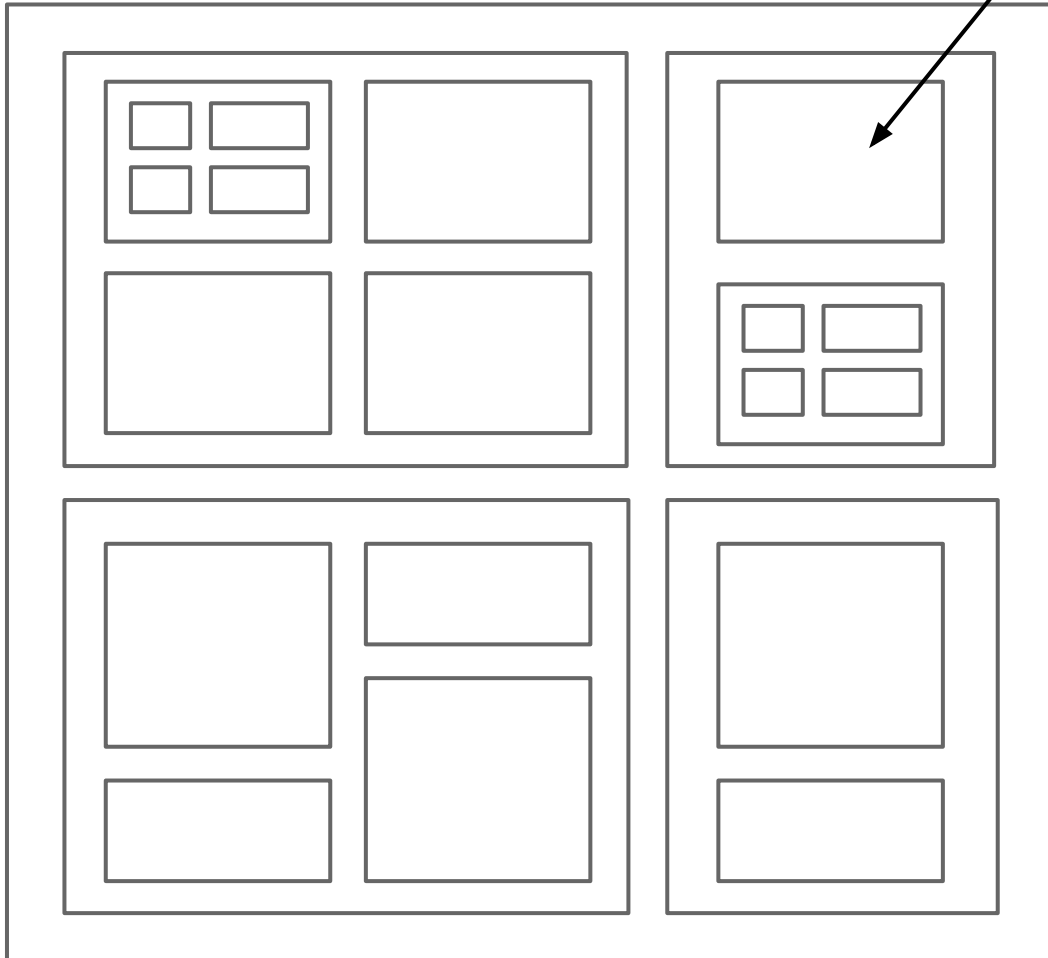


Процесс построения модели

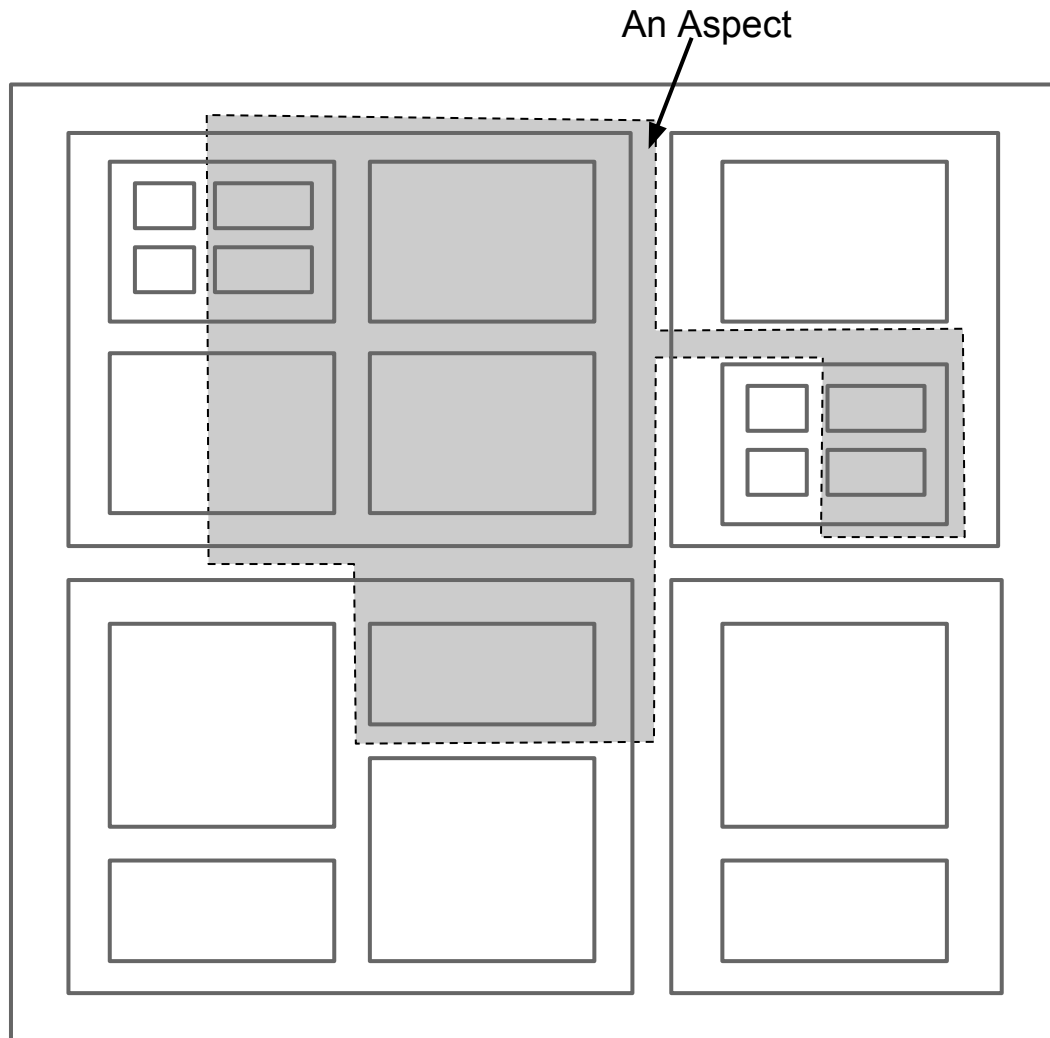
- Как определить структуру модели
 - Заинтересованные лица (пользователи)
 - Эксперты в данной области
 - Уже работающие системы
 - Примеры
 - Концепты и фичи
 - основные термины в терминологии пользовательской области - концепты (обычно существительные)
 - переменные значения, отличия между терминами (сущностями), варианты
 - возможность обобщения

Modular decomposition

A Modular Unit



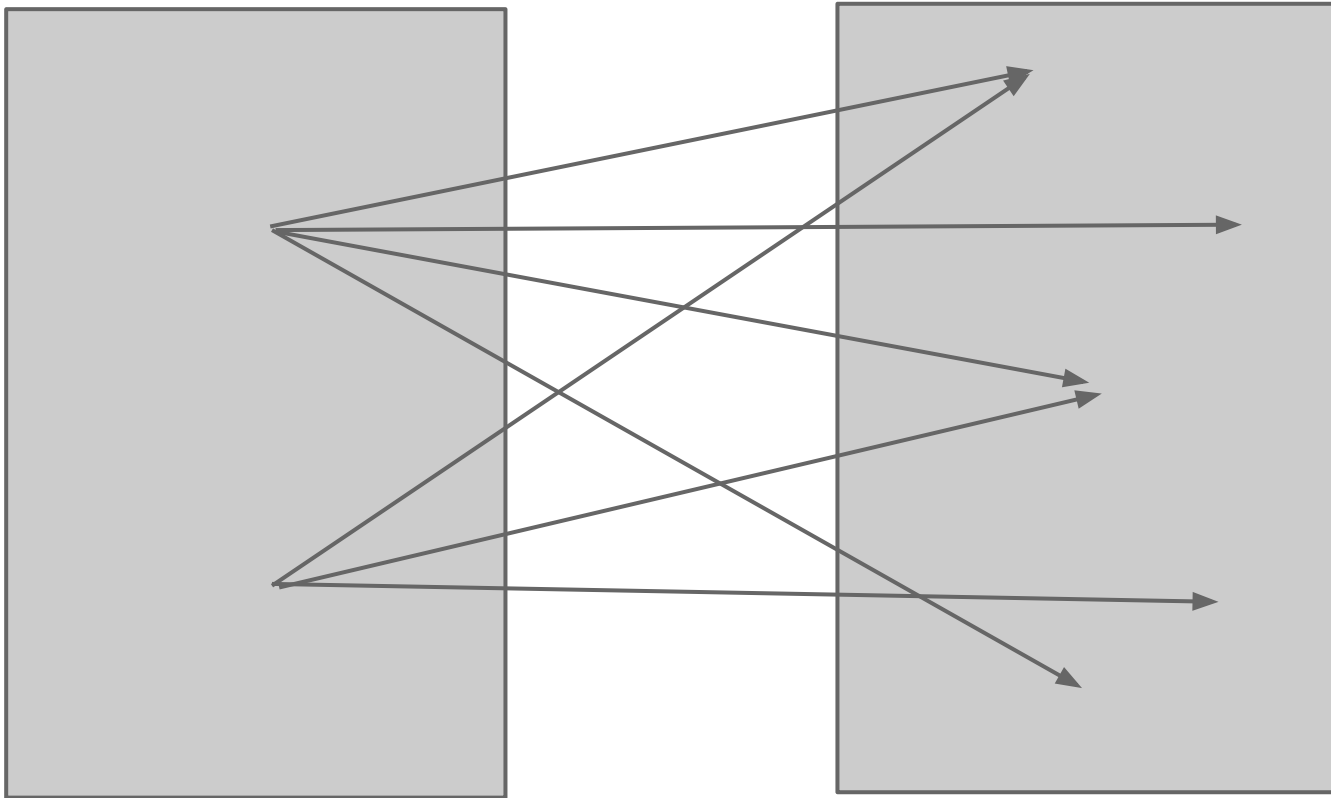
Aspectual decomposition



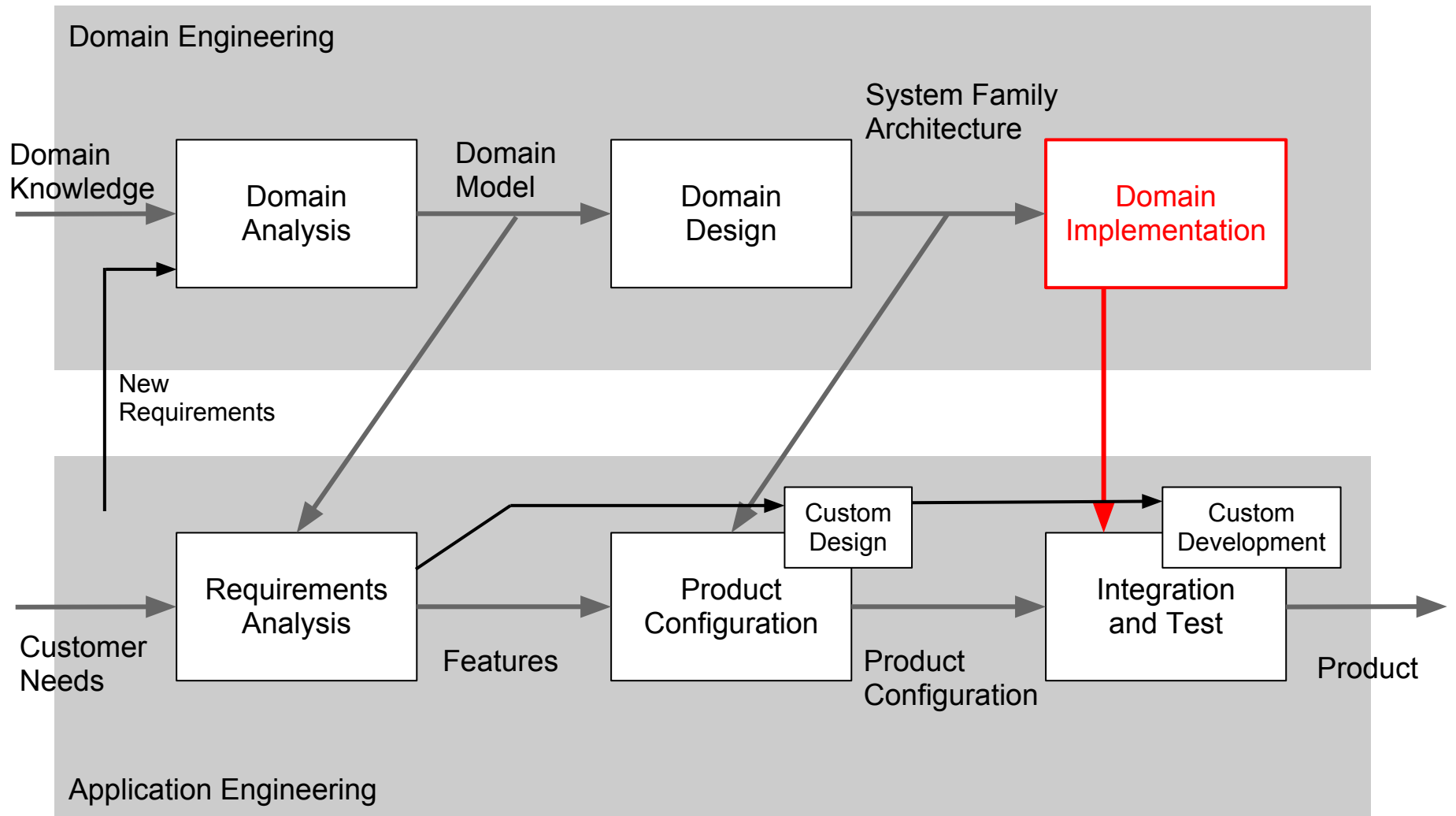
Aspectual decomposition

Model A

Model C



Domain Engineering



Generation

- Macro
- Text -> Text transformation
- Model -> Model transformation
- Model -> Text transformation

Macro

- simple macro

```
#define MY_COLOR 0x45678A
```

- parameterized macro

```
#define square(x) x*x
```

```
#define max(a, b) a>b ? a : b
```

Macro problems

square(a+b) -> a+b*a+b

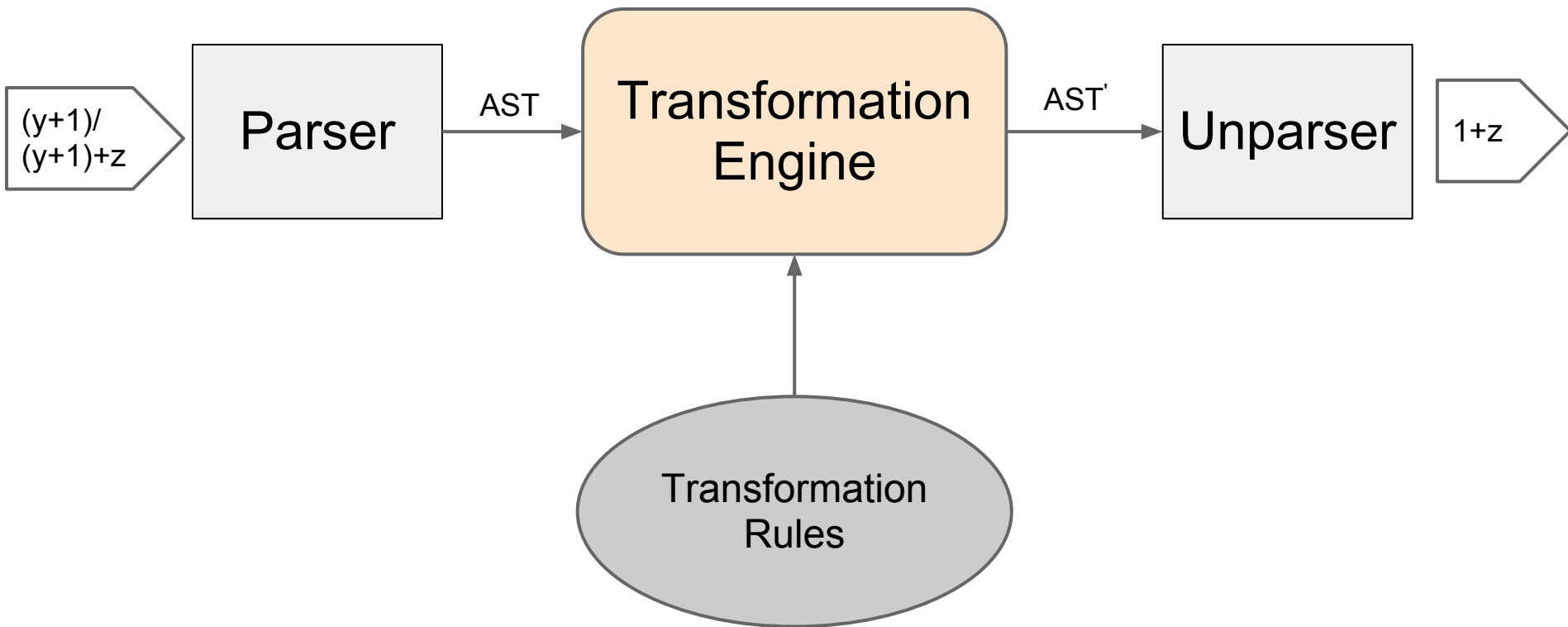
square(++a)

```
#define sum(a, len, result) \
```

```
    for (int i=0; i<len; ++i) result += a[i];
```

```
sum(a, 5, i);
```

Text to Text transformation



Пример XML-XSLT XSL - XSLT

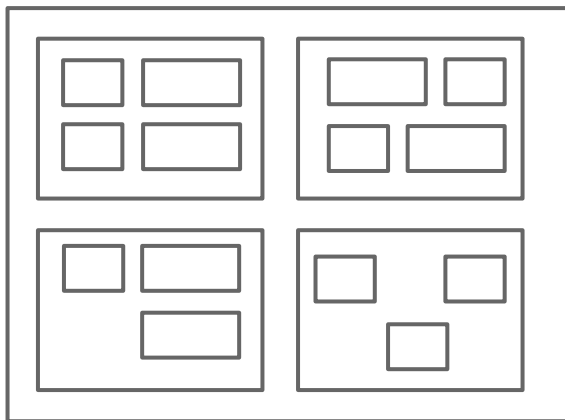
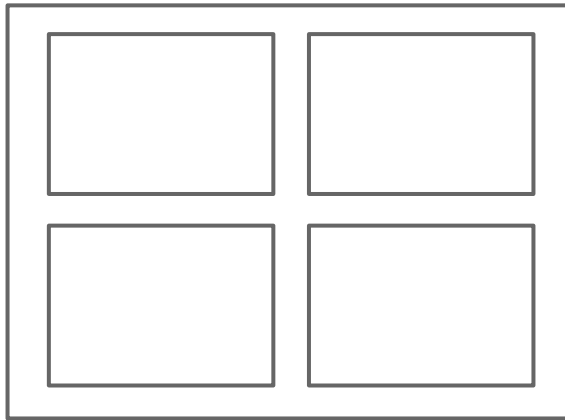
```
<?xml version="1.0"?>
<persons>
  <person username="MP123456">
    <name>Иван</name>
    <surname>Иванов</surname>
  </person>
  <person username="PK123456">
    <name>Пётр</name>
    <surname>Петров</surname>
  </person>
</persons>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<transform>
  <record>
    <username>MP123456</username>
    <fullname>Иван Иванов</fullname>
  </record>
  <record>
    <username>PK123456</username>
    <fullname>Пётр Петров</fullname>
  </record>
</transform>
```

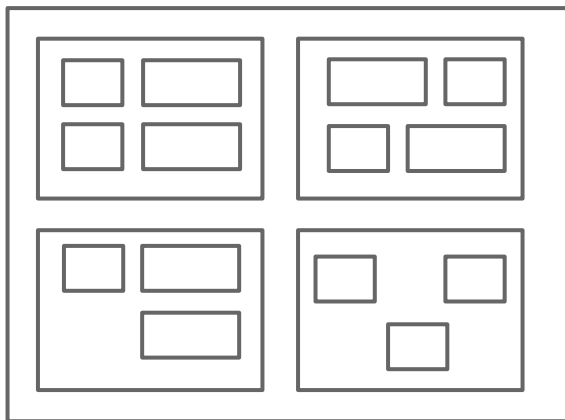
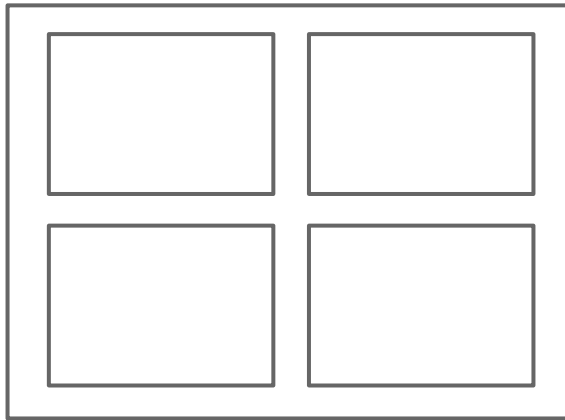
Пример XML-XSLT XSL - XSLT

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="xml" indent="yes"/>
  <xsl:template match="persons">
    <transform>
      <xsl:apply-templates/>
    </transform>
  </xsl:template>
  <xsl:template match="person">
    <record>
      <xsl:apply-templates select="@*|*" />
    </record>
  </xsl:template>
  <xsl:template match="@username">
    <username>
      <xsl:value-of select="."/>
    </username>
  </xsl:template>
  <xsl:template match="name">
    <fullname>
      <xsl:apply-templates/>
      <xsl:apply-templates select="following-sibling::surname" mode="fullname" />
    </fullname>
  </xsl:template>
  <xsl:template match="surname" />
  <xsl:template match="surname" mode="fullname">
    <xsl:text> </xsl:text>
    <xsl:apply-templates/>
  </xsl:template>
</xsl:stylesheet>
```


Vertical transformation (refinement)

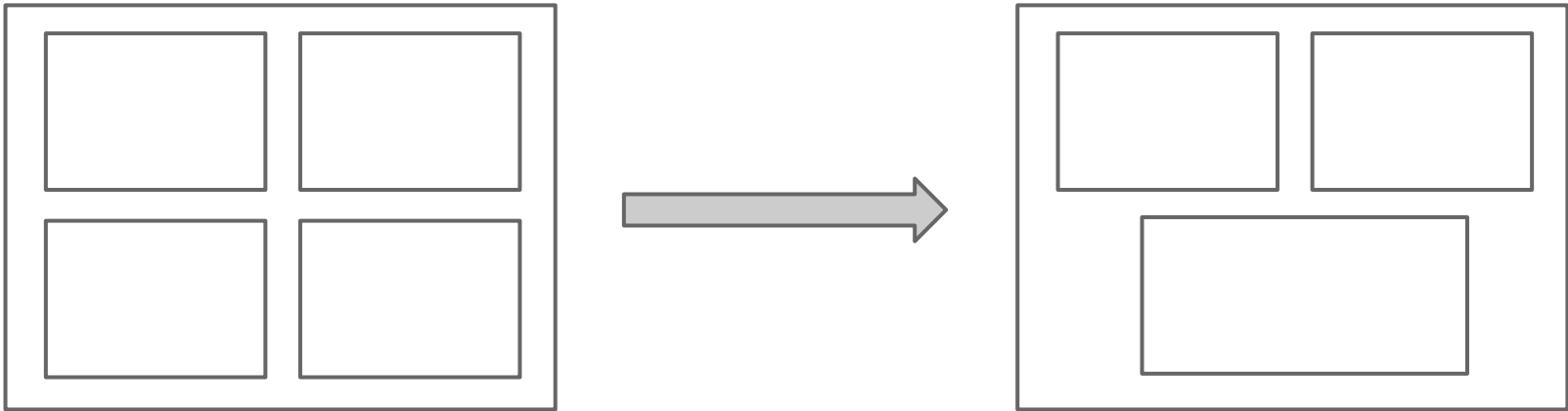


Vertical transformation (refinement)

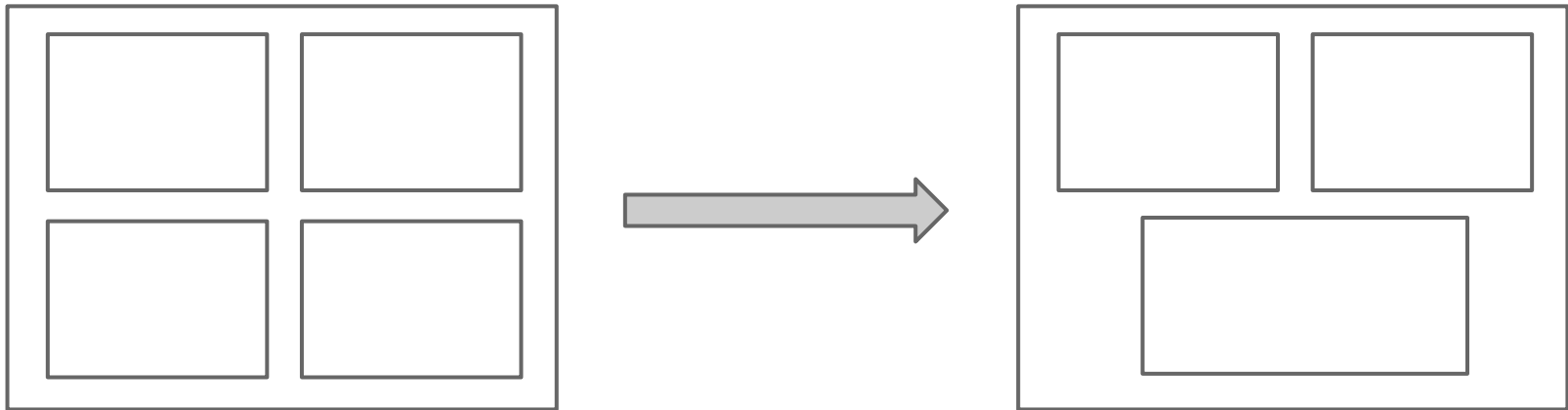


Развертывание
Выбор представления
Выбор алгоритма
Специализация

Horizontal transformation



Horizontal transformation



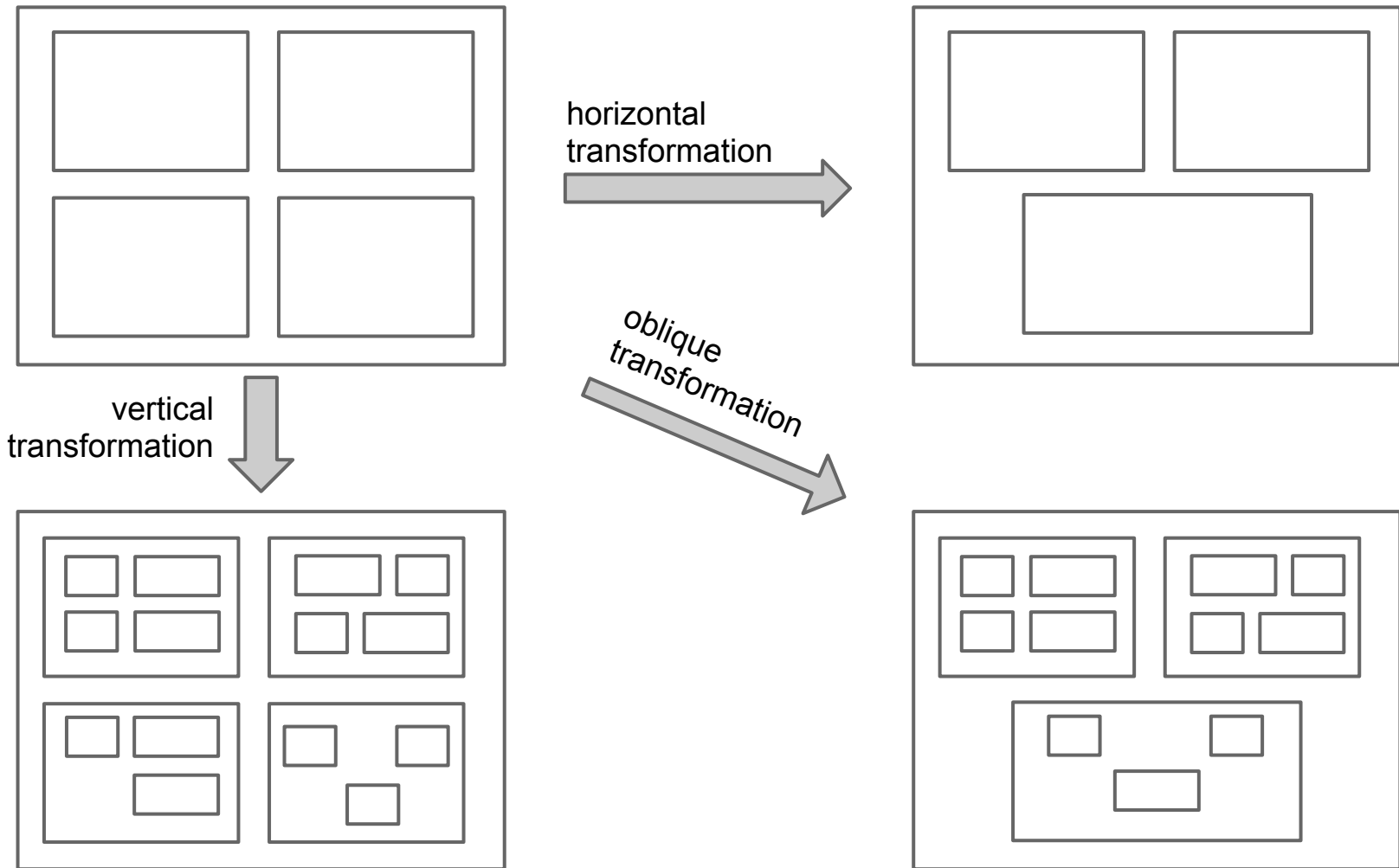
Оптимизации:

объединение нескольких концептов
распространение с верхнего уровня на
нижние (дочерние узлы получают
уточнения)

Примеры оптимизаций

- Подстановка (inlining)
- Вычисление констант (constant folding)
- Кеширование
- Объединение циклов
- Развертывание цикла
- Выделение инварианта цикла
- ...
- Специфичные для домена оптимизации
 - $a^x a^y == a^{x+y}$, $a^1 == a$
- Глобальные оптимизации

Transformations



MPS - generator

- модуль генератора
- конфигурации (mapping configuration) и приоритеты
- правила
- шаблоны и макросы
- скрипты (preprocess, postprocess)
- метки и ссылки (внутри модели)

MPS - generator

правила

root mapping rule

conditional root rule

reduction rule

pattern rule

abandon root rule

weaving rule

MPS - generator

макросы

property macro

reference macro

COPY_SRC

COPY_SRCL

IF

LOOP

SWITCH

CALL

MAP_SRC

Примеры

Литература

- Generative Programming: Methods, Tools, and Applications
Krzysztof Czarnecki, Ulrich Eisenecker
Addison-Wesley Professional; 1 edition (June 16, 2000)
Перевод: Чернецки К., Айзенекер У. Порождающее программирование: методы, инструменты, применение / Пер. с англ. СПб: Питер, 2005.
- Domain-Specific Languages
Martin Fowler
Addison-Wesley; 2011