Product Family Algebra

Vincent J. Maccio

Department of Computing and Software
McMaster University
Hamilton, Ontario, Canada

Outline

- What are product families, why use them
- Graphical methods
- Product family algebra
- Application
- Refinement

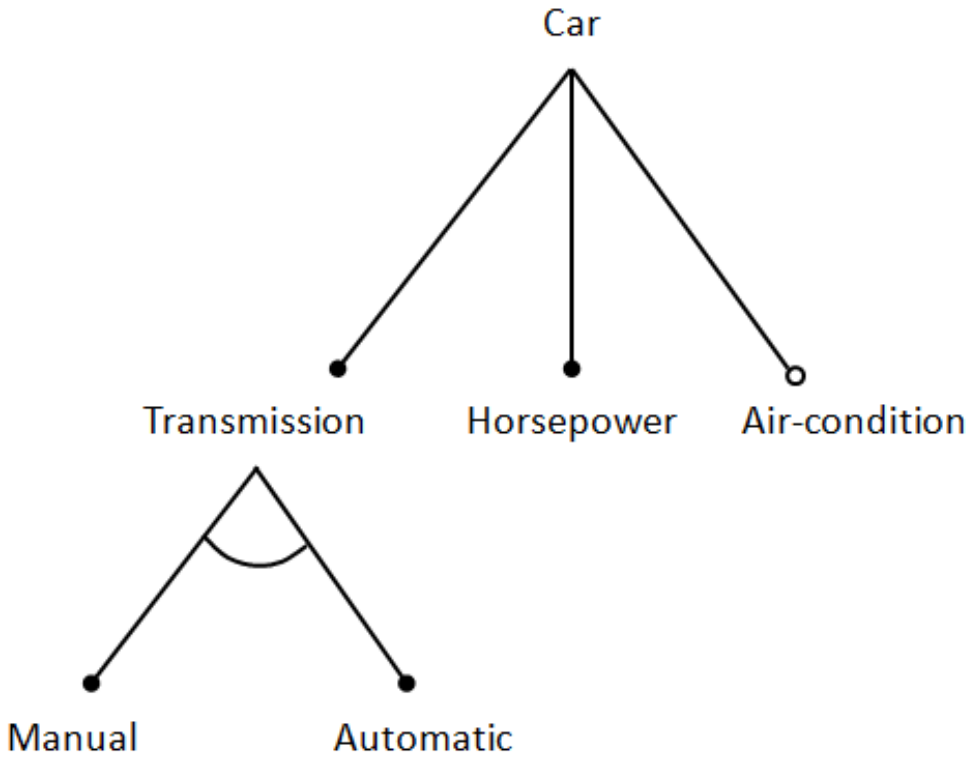
What is a Product Family

- A group of products that share a common set of features
- For example, phones, cars, houses, computers, etc.
- In our context (software engineering), common hardware or software artefacts (requirements, architectural properties, code)
- Such artefact are called features

Why use Product Families

- Reuse-ability of previous work (save resources, economies of scale, less overhead)
- Keeps your product flexible (different price points, different functionalities)
- Keeps your product competitive (consumers expect choices in many markets)
- In software, many of the principles of product families can be related to principles of good design

FODA



Looks good...

FORM

- Extends FODA with an emphasis on reuse-ability
- Separates features into four different layers (Capability, Operating, Domain Technology, Implementation Technique)
- More formalized methods for deriving/determining what a product "is"

Graphical Short Comings

Convenience Features	LX	HF	EX	EX-L
Power moonroof	--	--	●	●
Automatic temperature control	--	--	●	●
Air conditioning	●	●	●	●
Rear beverage holders	○	--	○	●
Speed control	●	●	●	●
Power front windows	●	●	●	●
Remote keyless entry	●	●	●	●
Tilt steering wheel	●	●	●	●
Telescoping steering wheel	●	●	●	●
Integrated navigation system	--	--	$1,500	$1,500
Front beverage holders	●	●	●	●

For a single car model from a single brand, there is over 100 (visible) features to the consumer.

Graphical Short Comings

- Soft complexity limitations
- Cannot perform calculus on them
- Some ambiguity exists, not rigorously formal
- Hard to switch between them

Product Family Algebra - Build Up

Monoids

The three-tuple $(S, \cdot, e)$ is a monoid if it satisfies the following axioms:

$$(\forall s \mid s \in S \ : \ e \cdot s = s \cdot e = s)$$
$$(\forall s_1, s_2, s_3 \mid s_1, s_2, s_3 \in S \ : \ s_1 \cdot s_2 \cdot s_3 = s_1 \cdot (s_2 \cdot s_3))$$

Commutative Monoids

The three-tuple $(S, \cdot, e)$ is a commutative monoid if $(S, \cdot, e)$ is a monoid and:

$$(\forall s_1, s_2 \mid s_1, s_2 \in S \ : \ s_1 \cdot s_2 = s_2 \cdot s_1)$$

Product Family Algebra - Build Up

Semirings

The quintuple $(S, +, 0, \cdot, 1)$ is a semiring if $(S, +, 0)$ is a commutative monoid, $(S, \cdot, 1)$ is a monoid, and:

$$(\forall s_1, s_2, s_3 \mid s_1, s_2, s_3 \in S \ : \ s_1 \cdot (s_2 + s_3) = (s_1 \cdot s_2) + (s_1 \cdot s_3))$$

Furthermore $(S, +, 0, \cdot, 1)$ is a commutative-semiring if:

$$(\forall s_1, s_2 \mid s_1, s_2 \in S \ : \ s_1 \cdot s_2 = s_1 \cdot s_2)$$

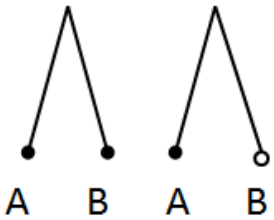
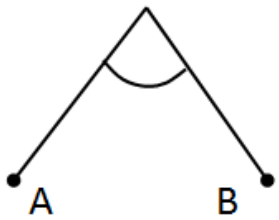
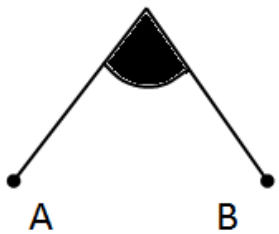
and $(S, +, 0, \cdot, 1)$ is an idempotent-semiring if:

$$(\forall s \mid s \in S \ : \ s + s = s)$$

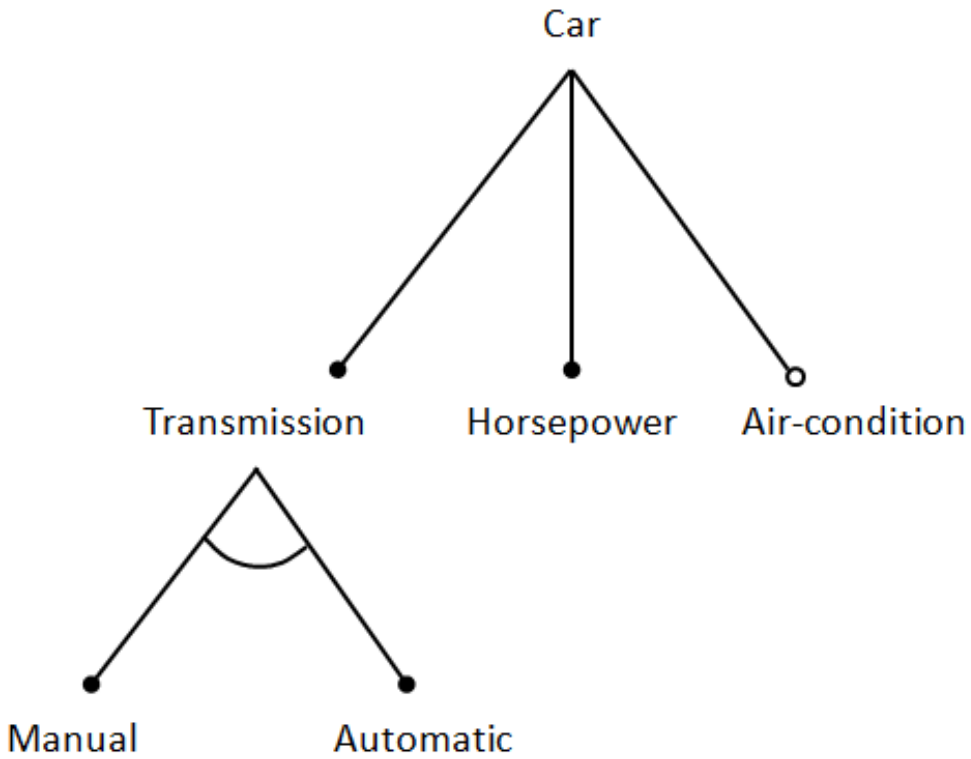
Interpretation

- The carrier set S is the set of families of products, where a product family can be viewed as a set of features related in certain ways
- 0 is the empty family of products, 1 is a family of a single product with no features
- $+$ is a choice between product families
- $\cdot$ is the composition of product families (mandatory features)
- Example $A \cdot (B + C) + D$
- Some added syntax: $opt[A] = (A + 1), opt[A, B] = (A + 1) \cdot (B + 1)$

FODA to PFA

	Mandatory and optional features	A, opt[A]
	Multiple features	A * B, A * opt[B]
	Alternative features	A + B
	Or-group	A + B + A*B

FODA to PFA



$$\text{horsepower} \cdot opt[A] \cdot (\text{manual} + \text{automatic})$$

Intuition

- $A + A = A, \quad A + B = B + A, \quad A + 0 = A$
- $A \cdot (B + 0) = A \cdot B + A \cdot 0 = A \cdot B$
- $A \cdot A = A$?

Set Model

- Let $\mathbb{F}$ be the set of all features
- A set of features is a product, the set of all possible products is the power set of features $\mathbb{P} = \mathcal{P}(\mathbb{F})$
- An element of the power set of $\mathcal{P}(\mathbb{P})$ is called a product family, and the power set itself is the set of all product families
- $1 = \{\emptyset\}$, $0 = \emptyset$

Set Model - Cont

- $\cdot \ : \ \mathcal{P}(\mathbb{P}) \times \mathcal{P}(\mathbb{P}) \rightarrow \mathcal{P}(\mathbb{P})$
- $P \cdot Q = \{p \cup q : p \in Q, q \in Q\}$
- $+ \ : \ \mathcal{P}(\mathbb{P}) \times \mathcal{P}(\mathbb{P}) \rightarrow \mathcal{P}(\mathbb{P})$
- $P + Q = P \cup Q$
- $\mathbb{P}FS = (\mathcal{P}(\mathbb{P}), +, \emptyset, \cdot, \{\emptyset\})$ forms a commutative idempotent semiring

Set Model - Example

- What is the carrier set if the set of all features $\mathbb{F} = \{a, b, c\}$?
- $\{\{a\}, \{a, b\}\} + \{\{b\}, \{a, b\}\}$?
- $\{\{a, b, c\}, \{a, b\}, \{a\}\} \cdot \{\{c\}, \{b\}\}$?

Bag Model

- $\cdot \ : \ \mathcal{P}(\mathbb{P}) \times \mathcal{P}(\mathbb{P}) \to \mathcal{P}(\mathbb{P})$
- $P \cdot Q = \{p \cup_{Bag} q : p \in Q, q \in Q\}$
- $+ \ : \ \mathcal{P}(\mathbb{P}) \times \mathcal{P}(\mathbb{P}) \to \mathcal{P}(\mathbb{P})$
- $P + Q = P \cup Q$
- $\mathbb{P}FS = (\mathcal{P}(\mathbb{P}), +, \emptyset, \cdot, \{\emptyset\})$ forms a commutative idempotent semiring

Refinement

- We wish to capture the notion that one product family "refines" another
- By refines we mean that all the products from one product family have at least all the features from at least one product from another product family
- Formally: $a \sqsubseteq b \Leftrightarrow \exists c : a \leq b \cdot c$
- Semi-Formally: $a \sqsubseteq b \Leftrightarrow (\forall \textit{products } p \in a : (\exists \textit{product } q \in b :$ p does everything q can do))

Results and Intuition

- $a \sqsubseteq a$?
- $a \sqsubseteq opt[b]$?
- $a \cdot opt[b] \sqsubseteq opt[a] \cdot b$?
- $a \cdot b \sqsubseteq a \cdot opt[b] + b \cdot opt[a]$?

Results and Intuition

- $a \leq b \Rightarrow a \sqsubseteq b$
- $a \cdot b \sqsubseteq b$
- $a \sqsubseteq a + b$
- $a \sqsubseteq b \Rightarrow a + c \sqsubseteq b + c$
- $a \sqsubseteq b \Rightarrow a \cdot c \sqsubseteq b \cdot c$
- $a \sqsubseteq 0 \Leftrightarrow a \leq 0$
- $0 \sqsubseteq a \sqsubseteq 1$

Requirements from Refines

- From the refines relation we can now begin to describe requirements
- Informally: $a \xrightarrow{e} b$ denotes if e has a it also has b
- Formally: $a \xrightarrow{p} b \Leftrightarrow p \sqsubseteq a \Rightarrow p \sqsubseteq b$

Conclusion

- Product families are a good way to describe product lines (software packages)
- PFA is a formal framework to describe and reason about these product families
- There are many applications and extensions on PFA making it a powerful tool

Thank you

Questions