what is a feature?

a qualitative study of features in industrial software product lines

Thorsten Berger, Daniela Lettner, Julia Rubin, Paul Grünbacher, Adeline Silva, Martin Becker, Marsha Chechik, Krzysztof Czarnecki
Professor, what is a feature?
A feature represents an aspect valuable to the customer. [Riebiesch03]

A Feature F of a product P is a product requirement $R \subseteq D$ that is visible to a user of the product P. [John10]

A feature is an increment of functionality, usually with a coherent purpose. [Zave99]

Customers and engineers usually speak of product characteristics in terms of the features the product has or delivers. [Kang++02]

we found 35 definitions of “feature”
features in industry

- when to introduce a feature?
- how to engineer a feature?
- what practices are effective in industry?
long-term goals

develop a model of what features are

design a feature prediction model

provide a more operational definition

“A journey of a thousand miles must begin with the first step.” [Laozi]
we qualitatively study real features

research questions

RQ1. What reasons cause companies to perceive a feature as typical, atypical, good or bad?
RQ2. What are important characteristics of features?

subjects

three companies, six interviewees

selected 23 features

typical, atypical (outlier), good, bad

interviews (~1.5h) and analysis guided by feature facets
20 feature facets guided interviews and analysis

description (rationale, nature, representation, position in hierarchy)

organization (level, responsibility)

implementation (scope, architectural responsibility, dependencies, implementation and deployment, inclusion/exclusion, binding time)

usage (use, evolution, metrics)

process (definition and approval, lifecycle purpose, testing, quality and performance)
SUBJECT COMPANIES
clone-and-own reuse ecosystem with internal and external developers
diverse feature representations
- product maps
- configuration tools
- code-level mechanisms
Opel (General Motors)

feature level

requirements level

Logical/physical architecture and deployment level

variability model: ~900 features (illustrative example)

C/C++ files
... #IFDEF TORQUE
... #IF defined(T1 & T2 )
... #ENDIF
... #ENDIF

integrated platform
1.5M lines of code
41,409 variation points (#IFDEFS)

configurator (pure::variants)

power-electronics firmware

Converter

Control Commands

Bus Communication

Compatibility

Torque

Wobbler

Cascade Controller

ResetFix

Converter

Control Commands

Bus Communication

Compatibility

Torque

Wobbler

Cascade Controller

ResetFix

Converter

Control Commands

Bus Communication

Compatibility

Torque

Wobbler

Cascade Controller

ResetFix
### Interviewees

<table>
<thead>
<tr>
<th>Part.</th>
<th>Role</th>
<th>Exp.</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>developer</td>
<td>12</td>
<td>LIN_Movement, Oscilloscope, Euromap, Silent_Mode</td>
</tr>
<tr>
<td>C</td>
<td>developer</td>
<td>3</td>
<td>Language_Translation, Production_Overview, DataManager, Heat-Up_Optimization</td>
</tr>
<tr>
<td></td>
<td><strong>Keba</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>team lead/architect</td>
<td>5</td>
<td>Lane_Keeping, Park_Assist, Emergency_Braking</td>
</tr>
<tr>
<td>E</td>
<td>architect</td>
<td>4</td>
<td>Torque, Cascade_Controller, Product_G, Power-Up_FastFuncs</td>
</tr>
<tr>
<td>F</td>
<td>team lead</td>
<td>8.5</td>
<td>Wobbler, Field_Bus, Reset_Fix, Board-Support_Package</td>
</tr>
<tr>
<td></td>
<td><strong>Opel</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Danfoss</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 participant (interviewee)  
2 experience with the product line in years

---

12
selection of

RESULTS
classification rationales

GOOD AND BAD FEATURES
individual features

good feature

- popular with customers
- popular with developers
- well implemented
- error-free
- thoroughly tested
- architecture-conform
- distinct functionality

bad feature

- customer complaints
- duplicate features
- workaround ("hack")
- defect features
- test challenges
- optional feature
- highly volatile
CROSS-CASE ANALYSIS

selection of results

using the 20 facets as a conceptual framework
distinct functionality

graspable / distinct features are good features

vague features are bad features

position in hierarchy
  interviewees preferred to talk about leaf or top-level features

What is problematic is when it’s too little specific.

Customers did not know what to expect.
outlier features

O1: Features do not only address functional or non-functional concerns that end up in a product. Features are also used for atypical purposes, such as supporting a system's lifecycle.

target dedicated lifecycle purpose (build, startup, QA)

incomplete process sufficient

restricted to some organizational levels

coordinated by subset of roles (responsibility)
examples of outliers

We didn’t really know how to improve it.

**Keba.UserGuidance**
placeholder for future functionality (usability improvement)

**Keba.HeatUpOptimization**
prevent startup power peaks

**Danfoss.BoardSupportPackage**
improve maintainability of Hardware Abstraction Layer

**Danfoss.PowerUpFastFuncs**
move functions from flash to RAM

We can tweak the product to indirectly fulfill the customer requirements.
features and parameters

O2: Parameters are not treated in the same way as features.

nature of features
   almost every feature came with configuration parameters

large parameter databases exist

parameters have
   no process attached
   no architectural responsibility
   no dedicated responsible role
cross-cutting features

O3: Scattered feature implementations do not necessarily lead to problematic features.

Scope of a feature is not a differentiator between good and bad half of the studied features were cross-cutting

There can be good reasons for a scattered feature implementation.

O5: Scattered features that have to be tested are problematic.

cross-cutting features problematic with manual testing processes only testable at integration time potentially require hardware

immature features

observed a diversity of different processes

process is not a differentiator between good and bad

but usually time pressure

Keba.ManualConfiguration

Danfoss’ time-boxing experiment

O4: A rushed development process causes problematic features.

There was an extreme pressure from the customer side.

We were told not to think, just to implement.
nature and use of features

features are primarily a unit of functionality
  used for communication among developers and customers
  used for scoping and creating awareness for software reuse

can serve as a unit of variability when necessary

Keba.DataManager
  introduced to provide low-level machine-data access
  making it optional caused significant effort

It rather felt like it’s a bug from the perspective of the customer.

It went back and forth: it’s a bug, it’s a feature, it’s a bug, it’s a feature; and then we said OK it’s a bug.

We need that for the [...] controlling.
we studied 23 real features
elicted key characteristics (facets)
studied good and bad practices
theory-building from cases
contributions for practitioners and researchers

**future work**
study feature lifecycles
create a model of what features are
design a prediction model
study other companies

what is a feature?
What is a feature?

Thorsten Berger, Daniela Lettner, Julia Rubin, Paul Grünbacher, Adeline Silva, Martin Becker, Marsha Chechik, Krzysztof Czarnecki