Freie Universität Bozen
Libera Università di Bolzano
Free University of Bozen - Bolzano

Fakultät für Informatik | Facoltà di Scienze e Tecnologie informatiche | Faculty of Computer Science

Feature Usage Diagram for Feature Reduction

Sarunas Marciuska, Cigdem Gencel, Xiaofeng Wang, and Pekka Abrahamsson

Outline

- Research Problem
- Related Work
- Feature Usage Diagram
- Case Study Design
- Results
- Conclusions
- Future Work

Research Problem

Feature creep [1] is adding features that have **no**, or **marginal value**.

Most of the software products contain **30 % - 50 %** unnecessary features [2].

- It makes a computer application slower
- Requires higher hardware capacities
- Increases the costs of maintenance

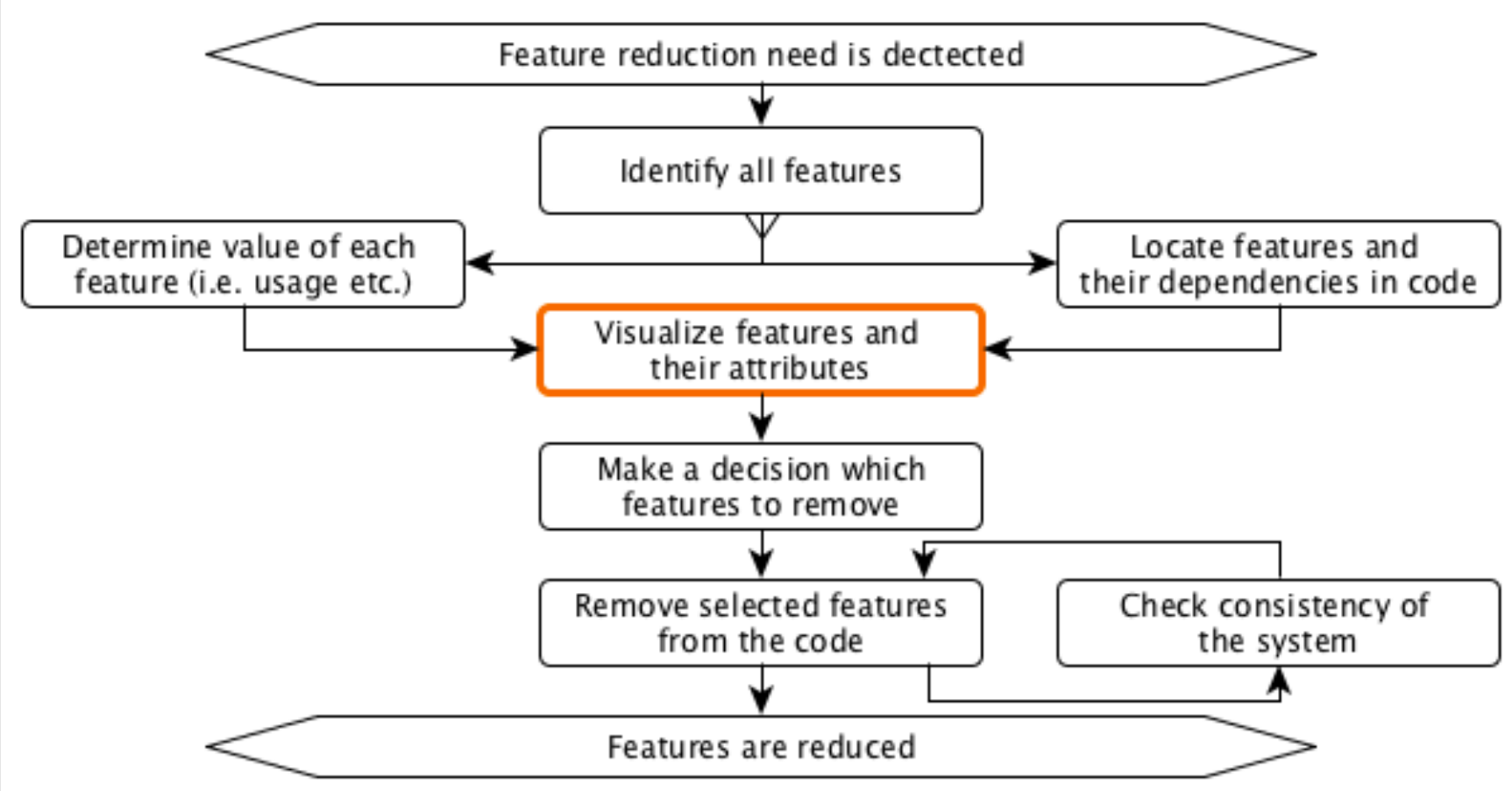
Current Approaches

Lean start-up [3] software development methodology tackles the feature creep problem by finding a minimum viable product.

Agile [4] software development practices focus on implementing valuable features by constantly considering changes in requirements.

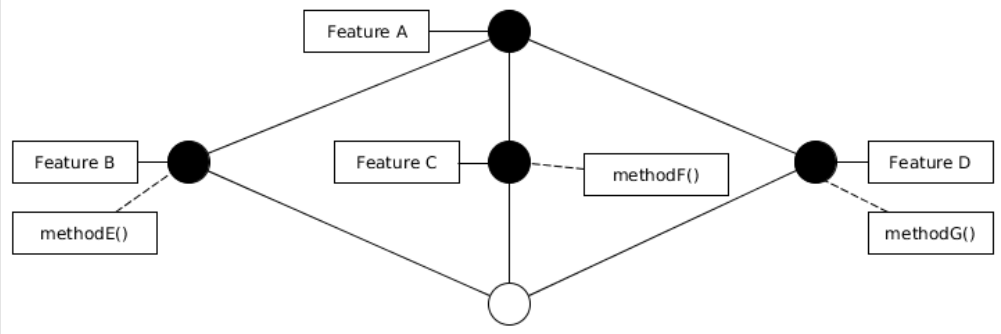
However, **it is difficult to determine the value** of a feature and find a minimal viable product. In addition, **some features loose their value over time** and might remain in a system unnoticed.

Feature Reduction

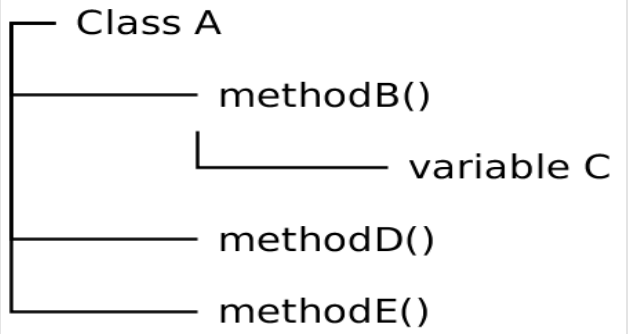


Related Work

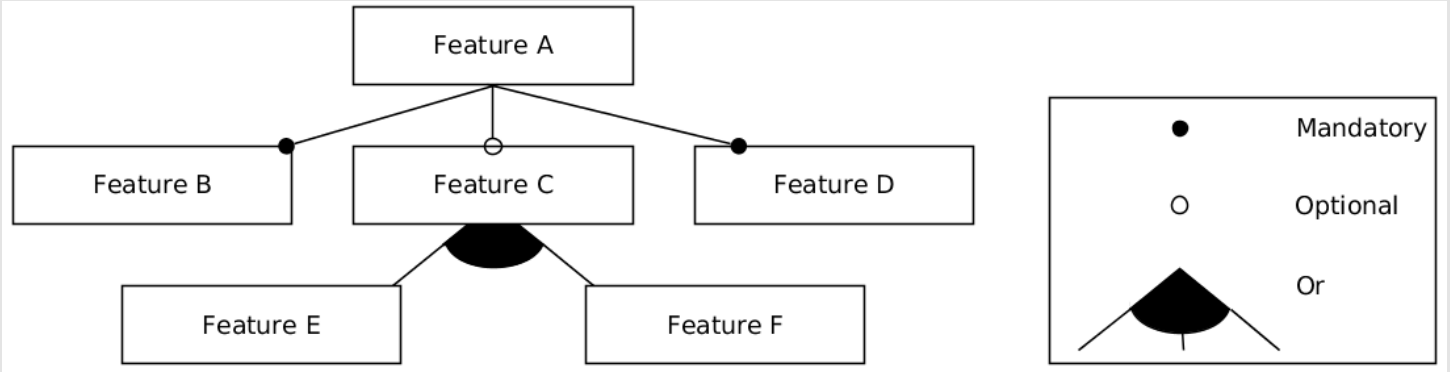
Concept lattice [5]



Concern Graph [6]



Feature diagram [7]

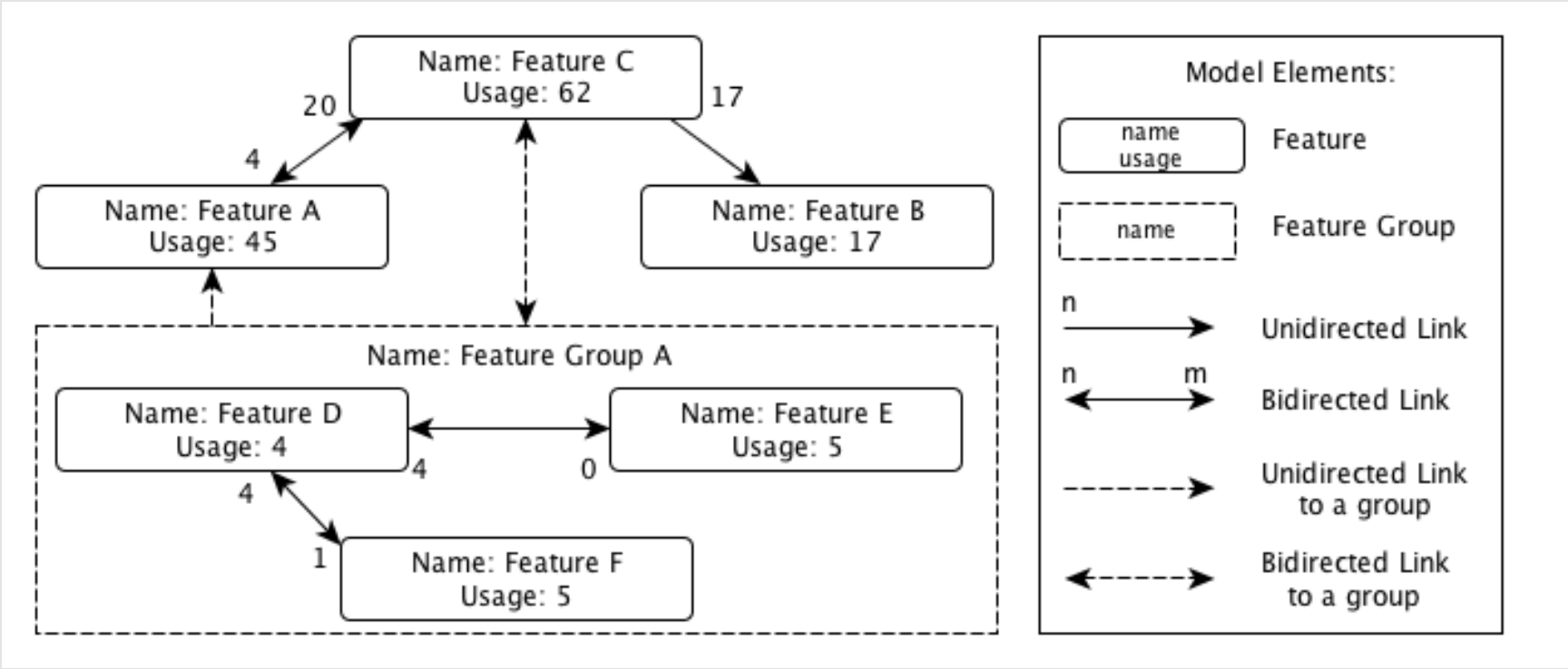


Related Work

State of the art methods do not capture value attributes related to features, nor they were created for this purpose.

To address this problem we designed a **Feature Usage Diagram** and Evaluated in the case study.

Feature Usage Diagram



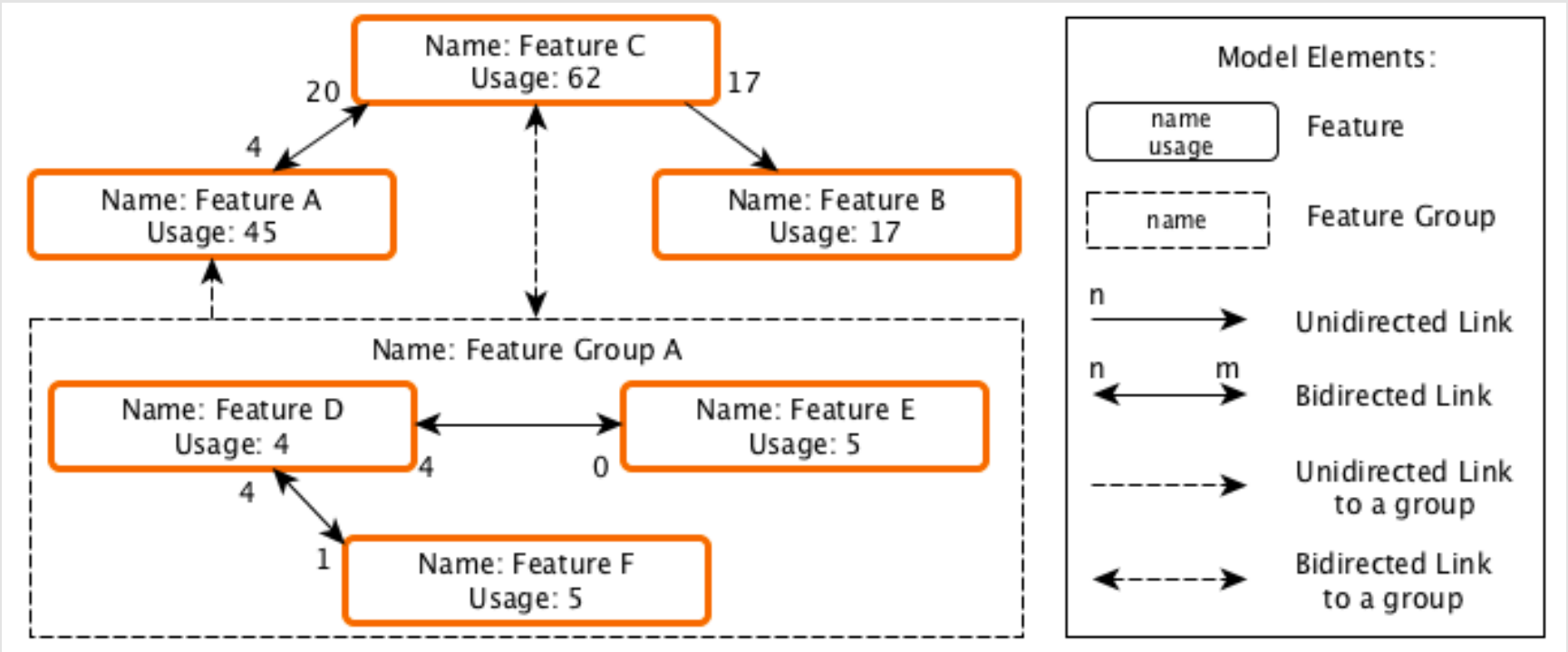
Feature

A **feature** is an observable unit of behaviour of a system triggered by the user [8]. Where user can be person, or other system.

Attributes:

- **Feature name** - represents a name of a feature.
- **Feature usage** - shows how many times a feature was used.

Feature


Name: Feature C
Usage: 62
20
17
4
Name: Feature A
Usage: 45
Name: Feature B
Usage: 17
Name: Feature Group A
Name: Feature D
Usage: 4
4
0
Name: Feature E
Usage: 5
4
1
Name: Feature F
Usage: 5
Model Elements:
name
usage
Feature
name
Feature Group
n
Unidirected Link
n
m
Bidirected Link
Unidirected Link
to a group
Bidirected Link
to a group

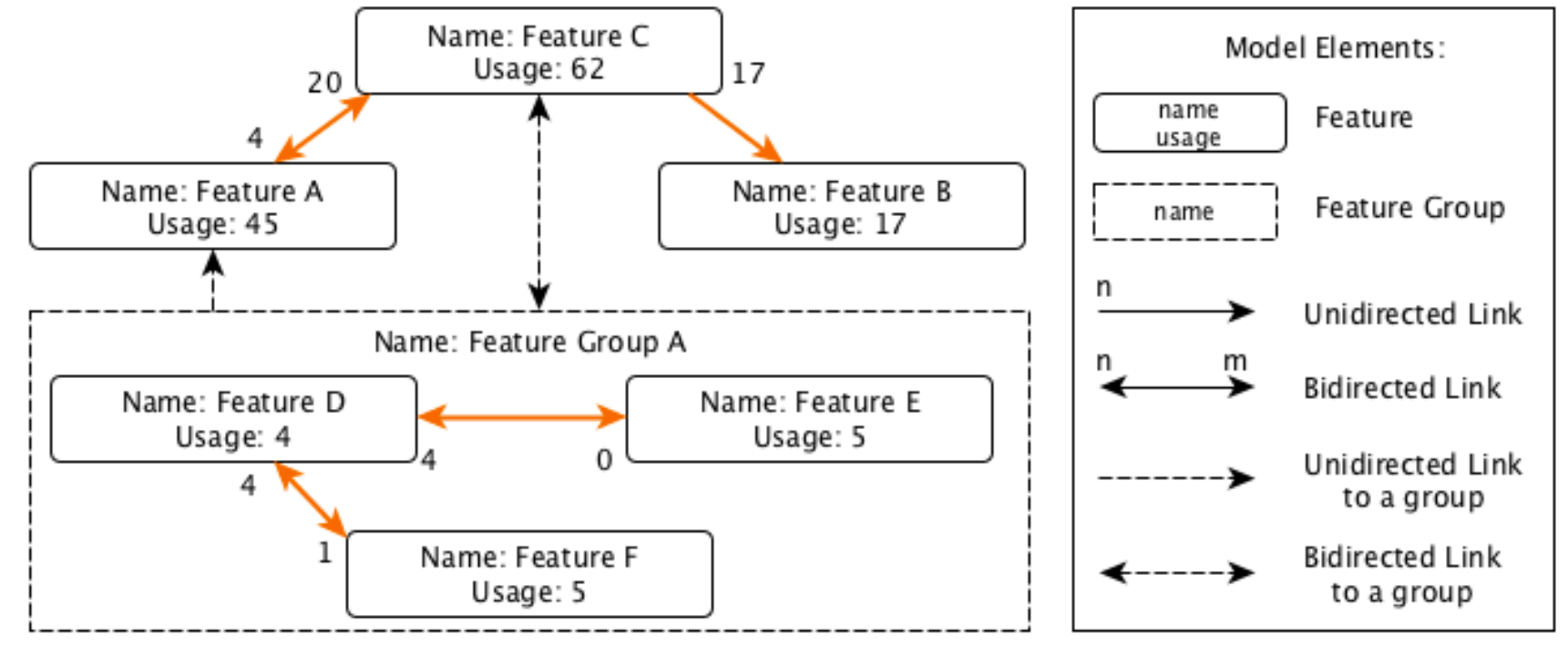
Directed Link Between Features

Directed Link Between Features represents an access path between two Features.

Attributes:

Link Cardinality - shows how many times a feature, which the link is pointing to, was accessed from a connected feature.

Directed Link Between Features



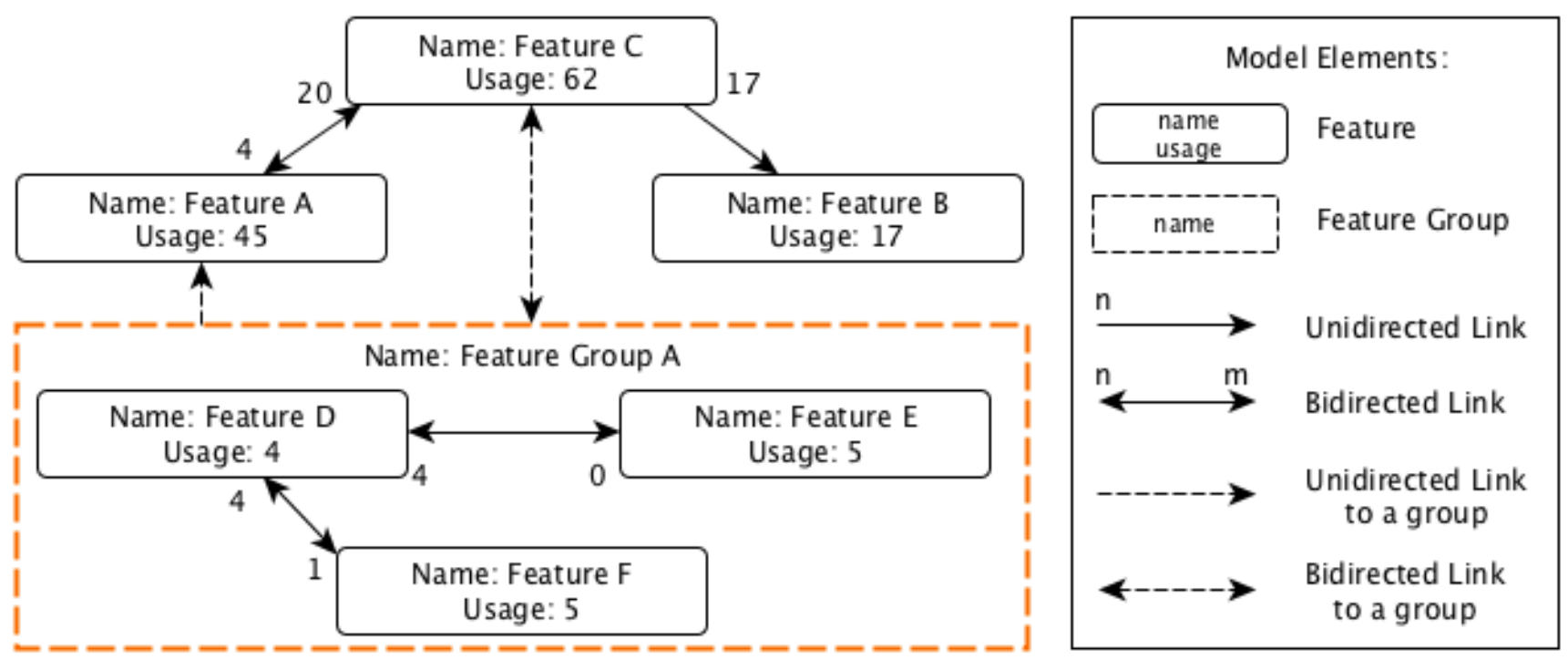
Feature Group

Feature Group element can be used to group Features that have the same links between the Features that are outside the group.

Attributes:

- **Group name** - represents a name of a feature group.

Feature Group



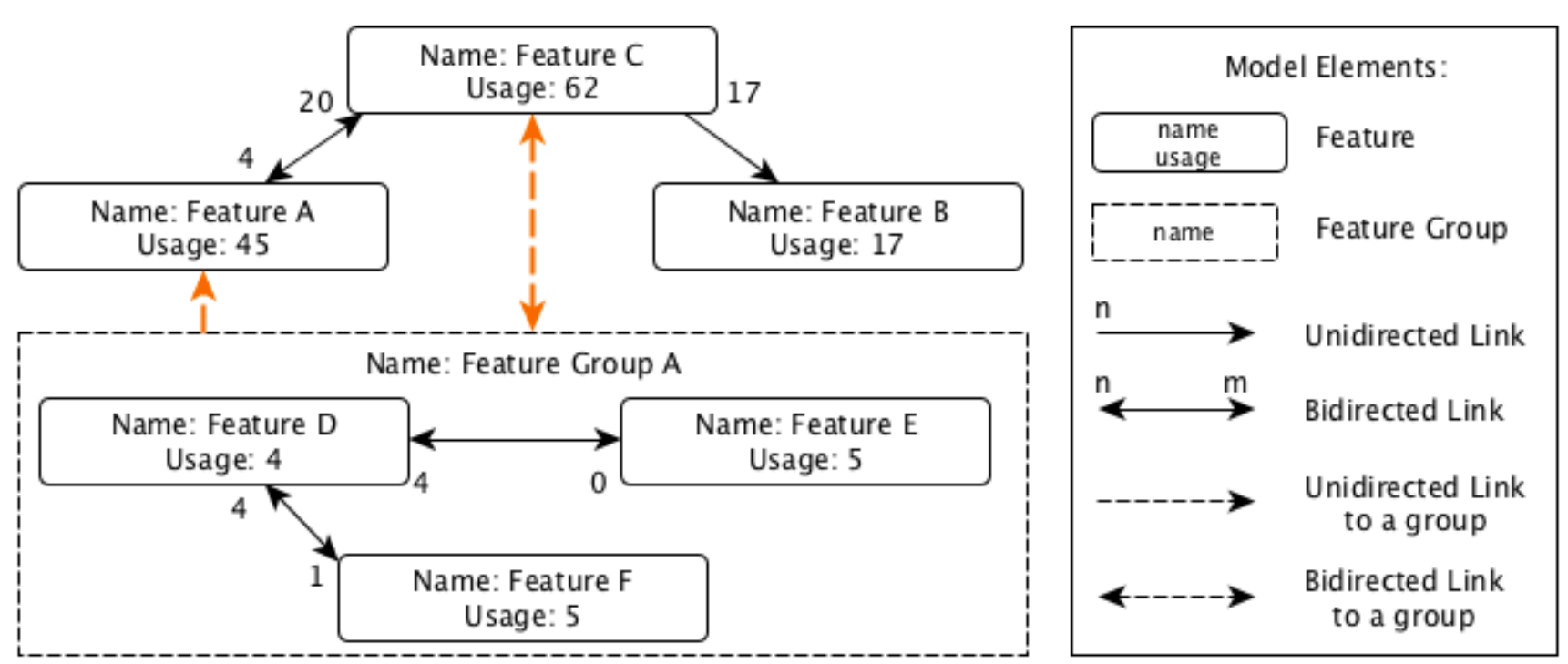
Directed Link to Feature Group

Directed Link to Feature Group shows how a feature group is connected to other feature groups, or features.

Attributes:

None.

Directed Link to Feature Group



Case Study Design

Case application: nextrailer.net web based movie recommender system. It had 30 Features and 200 Directed Links between them.

Research Questions:

- **RQ1**: Is the Feature Usage Diagram easy to learn and understandable to use?
- **RQ2:** Is the Feature Usage Diagram notation complete to represent features and their dependencies (nothing is redundant or missing)?
- **RQ3**: Is the visualised information on the Feature Usage Diagram useful for decision making in feature reduction?

Participants: 12 Computer Science students (MSc and PhD students) having programming an modeling experiences.

Case Study Conduct (Phase 1)

- Participants were introduced to Feature Usage Diagram.
- The participants modeled the case application using Google Drawing graphical editor.
- To understand if it is easy to learn the notation (**RQ1**) we compared the time spent on the task by each participant to the time spend by the first author of this paper.
- To investigate if notation is understandable (**RQ1**) we compared the results of the participants to a complete and correct version of the diagram provided by the developers of the system.
- We made an unstructured interview with each participant to understand if the notation is complete to represent features and their dependencies (**RQ2**).

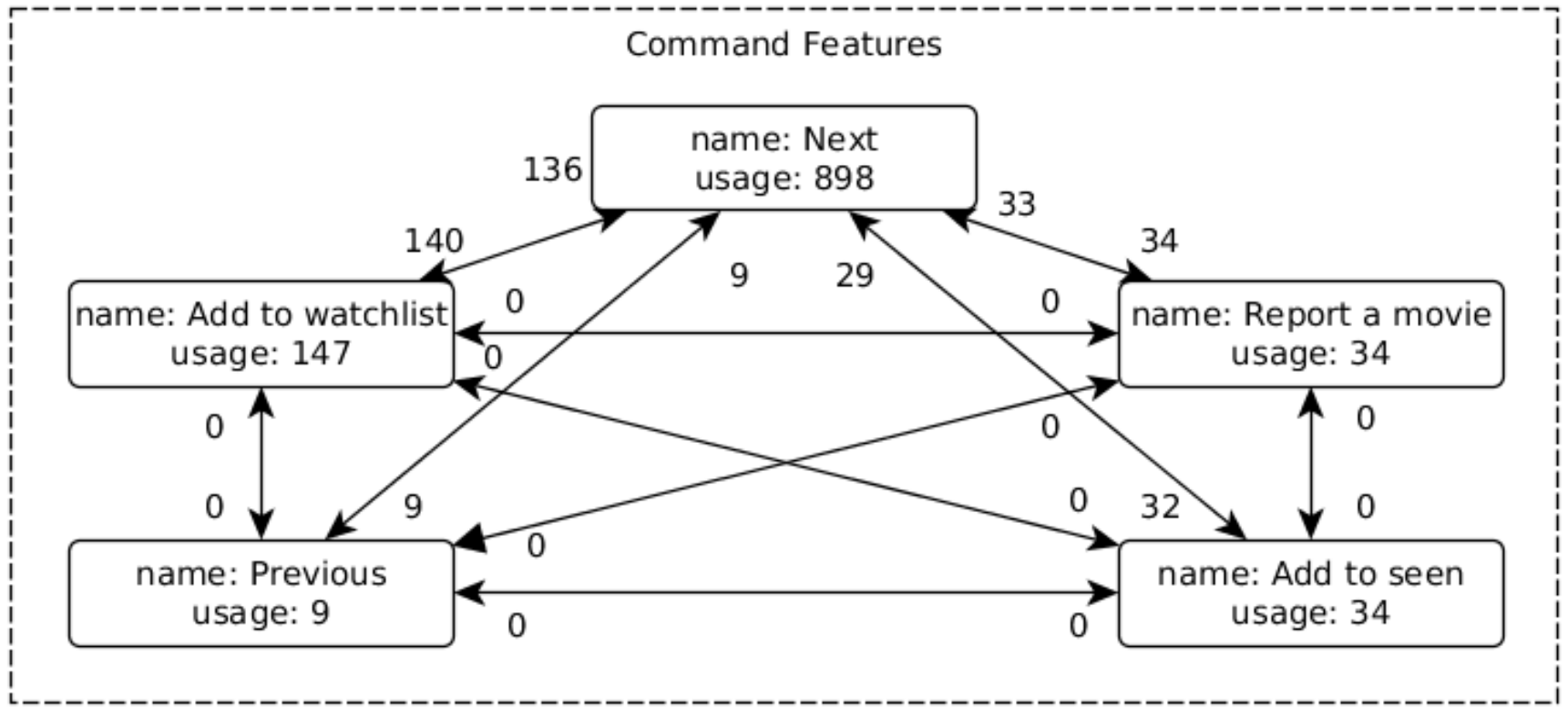
Case Study Conduct (Phase 2)

- We collected feature usage data on nextrailer.net website for 30 days.
- Then, we inserted this information to the correct version of Feature Usage Diagram.
- Finally, we interviewed the developers to understand if the visualised information on the Feature Usage Diagram is useful for decision making in feature reduction (**RQ3**).

Results (Phase 1)

- Feature Usage Diagram notation was easy to learn (**RQ1**), as the times to draw it were similar among participants and the first author of the paper.
- Feature Usage Diagram notation is understandable to use (**RQ1**), as most of the participant could identify feature and feature groups correctly.
- Interviews with the participant revealed that Cyclic Bi-directed Link on a Feature Group, would be useful extension for this notation (**RQ2**).
- However, participants noted that tool support for the notation would be useful.

Results (Phase 2)



Results (Phase 2)

- During the interviews, all developers pointed out that the Feature Usage Diagram would be very useful for decision making in feature reduction (**RQ3**).
- Moreover, usage information could be useful for value maximisation purposes in order to make decisions for how to modify a system, which generates more value.
- However, other aspects of value should be taken into account as well.

Conclusions

- We introduced a new diagram to visualise the features, their relationships, and their usage information.
- The Feature Usage Diagram elements and notation is easy to learn and understand by novice users.
- Feature Usage Diagram has a potential to aid developers in decision making for feature reduction purposes.

Future Work

- We plan to investigate other aspects of feature value and extend the Feature Usage Diagram to incorporate these aspects as well.
- We will explore how value can be maximized by relocating features in different places of a system.
- We will analyse the challenges faced to apply feature Usage Diagram to big systems with a high number of features.
- We aim to develop a tool to support for drawing Feature Usage Diagram.
- We will explore the ways how to automatically generate a the diagram.

References

[1] Davis, F.D., Venkatesh, V.: Toward preprototype user acceptance testing of new information systems: implications for software project management. In: IEEE Transactions on Engineering Management (2004).

[2] Ebert, C., Dumke, R.: Software Measurement. Springer (2007).

[3] Ries, E.: The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. In: Journal of Product Innovation Management (2011).

[4] Highsmith, J. A.: Agile software development ecosystems. Addison-Wesley Professional (2002).

[5] Poshyvanyk, D., Marcus, A.: Combining formal concept analysis with information retrieval for concept location in source code. In: Program Comprehension, pp. 37-48(2007).

[6] Robillard, M. P., Murphy, G. C.: Concern Graphs: Finding and describing concerns using structural program dependencies. In: Proceedings of International conference on software engineering, pp. 406-416 (2002).

[7] Benavides, D., Trinidad, P., and Ruiz-Cortes, A.: Automated reasoning on feature models. In: Advanced Information Systems Engineering, pp. 381-390 (2005).

[8] Eisenbarth, T., Koschke, R., Simon, D.: Locating Features in Source Code. In: IEEE Computer (2003).

Questions

Thank you for your attention!

Sarunas Marciuska
Free University of Bolzano
Marciuska@inf.unibz.it

Cigdem Gencel
Free University of Bolzano
Cigdem.Gencel@unibz.it

Xiaofeng Wang,
Free University of Bolzano
Xiaofeng.Wang@unibz.it

Pekka Abrahamsson
Free University of Bolzano
Pekka.Abrahamsson@unibz.it