# Exploring the 3-Dimensional Variability of Websites' User-Stories using Triadic Concept Analysis

Alexandre Bazin[1], Thomas Georges[1,3], Marianne Huchard[1], Pierre Martin[2], Chouki Tibermacine[1]

[1]LIRMM, Université de Montpellier, CNRS
[2]AIDA, Cirad
[3]ITK -Predict & Decide

CONCEPTS'24

# Context

## Software Product Line (SPL)

- Framework to build similar software in a disciplined way
- Low cost, easy customization
- Various implementation approaches
- Main common artefact: Variability model (options/features and constraints)

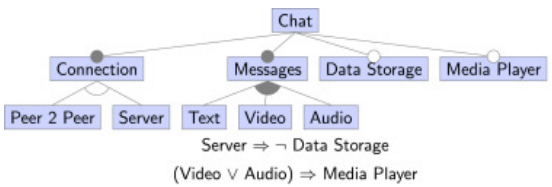
## Extractive approach to build a SPL

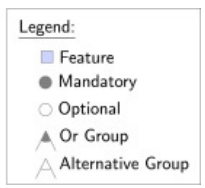
- Using existing similar software
- Extract Variability model, shared artefacts (requirements, specification, code)

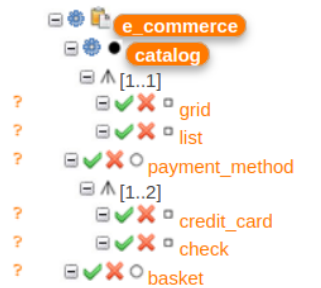
# Context

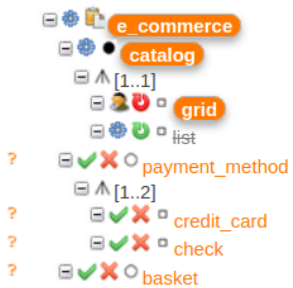
## Variability model

- Available features and constraints
- Partial or exhaustive expression of variability
- Diverse forms: Feature models, Textual models (UVL), CSP, Binary Implication Graph, Variable UML diagrams



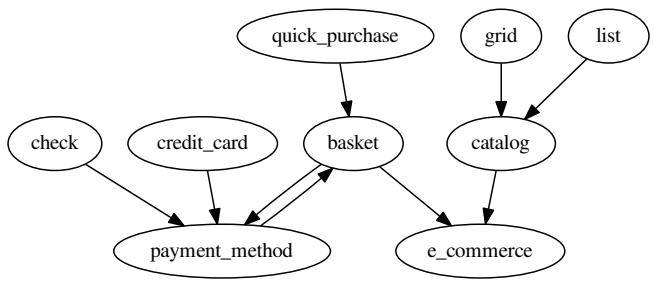



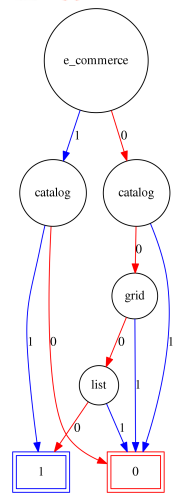




```
features
    Chat
        mandatory
            Connection
                alternative
                    "Peer 2 Peer"
                    Server
            Messages
                or
                    Text
                    Video
                    Audio
        optional
            "Data Storage"
            "Media Player"

constraints
    Server => "Data Storage"
    Video | Audio => "Media Player"
```



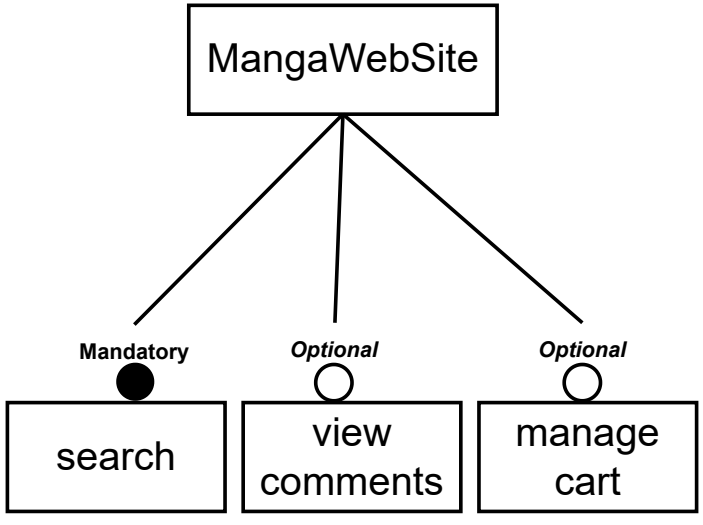

Credits: J. Galasso and https://www.sciencedirect.com/science/article/pii/S016412122400195X

# Extract variability model

## Standard case

### Software described by features

| | $search\ (s)$ | $view\ comment\ (vc)$ | $manage\ cart\ (mc)$ |
|---|---|---|---|
| $MyManga$ | $\times$ | | $\times$ |
| $MangaStore$ | $\times$ | | |
| $MangaHome$ | $\times$ | $\times$ | $\times$ |




$\{\} \Rightarrow$ search

view comment, search $\Rightarrow$ manage cart

*Graphical variability expression (e.g. feature model [Kang, 1990])*

*Textual variability expression (e.g. dyadic implications)*

# Extract Software-feature-role variability model

## Agile approach

- Software described by a set of user-stories
- *As a* $<$ *user/role* $>$*, I want to* $<$ *action/feature* $>$

| | $s$ | $vc$ | $mc$ | $s$ | $vc$ | $mc$ | $s$ | $vc$ | $mc$ |
|---|---|---|---|---|---|---|---|---|---|
| $MyManga$ | $\times$ | | | $\times$ | | $\times$ | | | $\times$ |
| $MangaStore$ | $\times$ | | | $\times$ | | | | | |
| $MangaHome$ | $\times$ | $\times$ | | | $\times$ | | | $\times$ | $\times$ |
| | $FinalUser$ | | | $Administrator$ | | | $ProductManager$ | | |

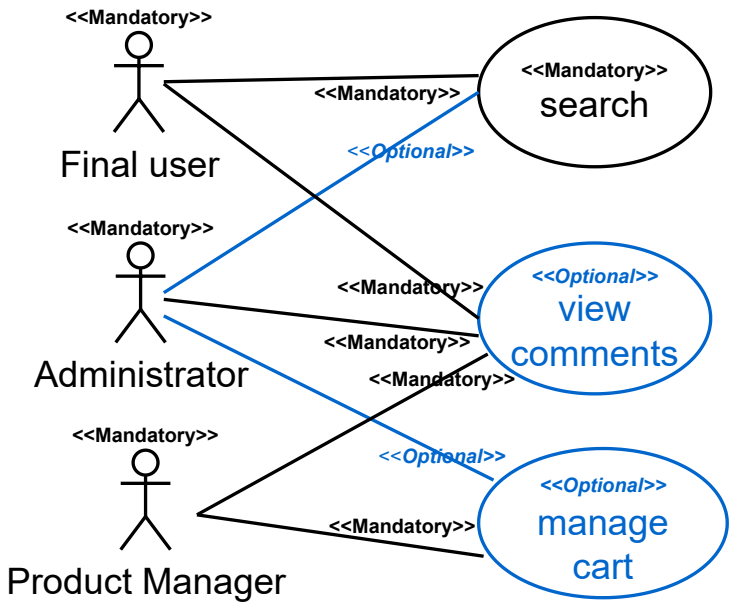
## Interest

- For software with various access (e.g. e-shop, forum, ERP, wiki, healthcare software, etc.)
- To take into account the role dimension in variability and further in new software configurations and code generation

# Extract Software-feature-role variability model

## State of the art

- No known extraction method
- No identified representation: Variable use case diagram in UML [Junior et al., 2010])?, logical expressions?




*Graphical variability expression (freely inspired from SMarty profile [Junior et al., 2010])*

$\forall (S, F) \in C_{FinalUser}$
$\cap\, C_{ProductManager},$
$(S, F) \in C_{Administrator}$

In $C_{FinalUser}$,
$viewComment \Rightarrow search$

*Possible textual variability expression*

# Proposed solution for extraction: Triadic Concept Analysis

Ternary relations between three object sets: software, roles and features

| | $s$ | $vc$ | $mc$ | $s$ | $vc$ | $mc$ | $s$ | $vc$ | $mc$ |
|---|---|---|---|---|---|---|---|---|---|
| $MyManga$ | × | | | × | | × | | | × |
| $MangaStore$ | × | | | × | | | | | |
| $MangaHome$ | × | × | | | × | | | × | × |
| | $FinalUser$ | | | $Administrator$ | | | $ProductManager$ | | |

# Triadic implications

What is a rule in a triadic setting?

- Rules between software
- Rules between roles
- Rules between features
- Rules between pairs (feature, role)
- Rules between pairs (software, feature)
- Rules between pairs (software, role)
- Rules that hold only given some condition

# Triadic implications: Conditioning

| | $s$ | $vc$ | $mc$ | $s$ | $vc$ | $mc$ | $s$ | $vc$ | $mc$ |
|---|---|---|---|---|---|---|---|---|---|
| $MyManga$ | $\times$ | | | $\times$ | | $\times$ | | | $\times$ |
| $MangaStore$ | $\times$ | | | $\times$ | | | | | |
| $MangaHome$ | $\times$ | $\times$ | | | $\times$ | | | $\times$ | $\times$ |
| | $FinalUser$ | | | $Administrator$ | | | $ProductManager$ | | |

$$\mathcal{C}_{\{FinalUser\}}$$

| | $search$ | $viewComment$ | $manageCart$ |
|---|---|---|---|
| $MyManga$ | $\times$ | | |
| $MangaStore$ | $\times$ | | |
| $MangaHome$ | $\times$ | $\times$ | |

Rule that holds only given some condition

$$(\{viewComment\} \Rightarrow \{search\})_{\{FinalUser\}}$$

# Triadic implications: Rules between Pairs (feature,role)

| | $(s,FU)$ | $(vc,FU)$ | $(mc,FU)$ | $(s,A)$ | $(vc,A)$ | $(mc,A)$ | $(s,PM)$ | $(vc,PM)$ | $(mc,PM)$ |
|---|---|---|---|---|---|---|---|---|---|
| $MyManga$ | × | | | × | | × | | | |
| $MangaStore$ | × | | | × | | | | | |
| $MangaHome$ | × | × | | | × | | | × | × |

$$\{(viewComment, Administrator)\} \Rightarrow \{(manageCart, ProductManager)\}$$

# Triadic implications: Rule between Roles

| | $FinalUser$ | $Administrator$ | $ProductManager$ |
|---|---|---|---|
| $(MM,s)$ | × | × | |
| $(MM,vc)$ | | | |
| $(MM,mc)$ | | × | × |
| $(MS,s)$ | × | × | |
| $(MS,vc)$ | | | |
| $(MS,mc)$ | | | |
| $(MH,s)$ | × | | |
| $(MH,vc)$ | × | × | × |
| $(MH,mc)$ | | | × |

$$\{FinalUser, ProductManager\} \Rightarrow \{Administrator\}$$

# Case Study

## Goal

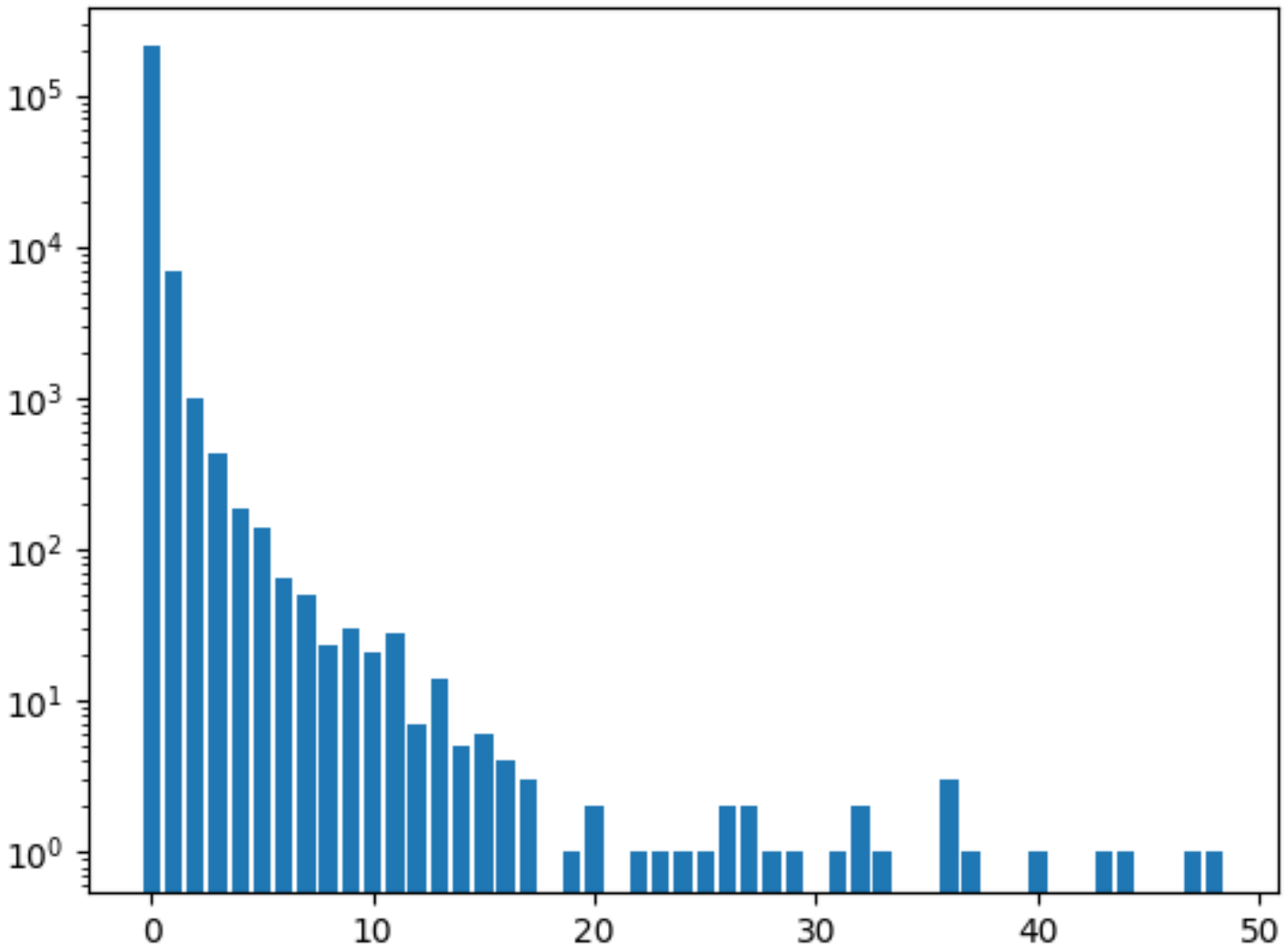
- An help for a software engineer:
  - Reasonable number of implications after relevant filtering
  - Understandable implications
  - Informative implications

## Dataset: Manga Websites

- 67 systems
- 145 features
- 17 roles
- 1546 triples (density 0.009)

# Case Study

## Number of Rules between Features per Support



# Case Study

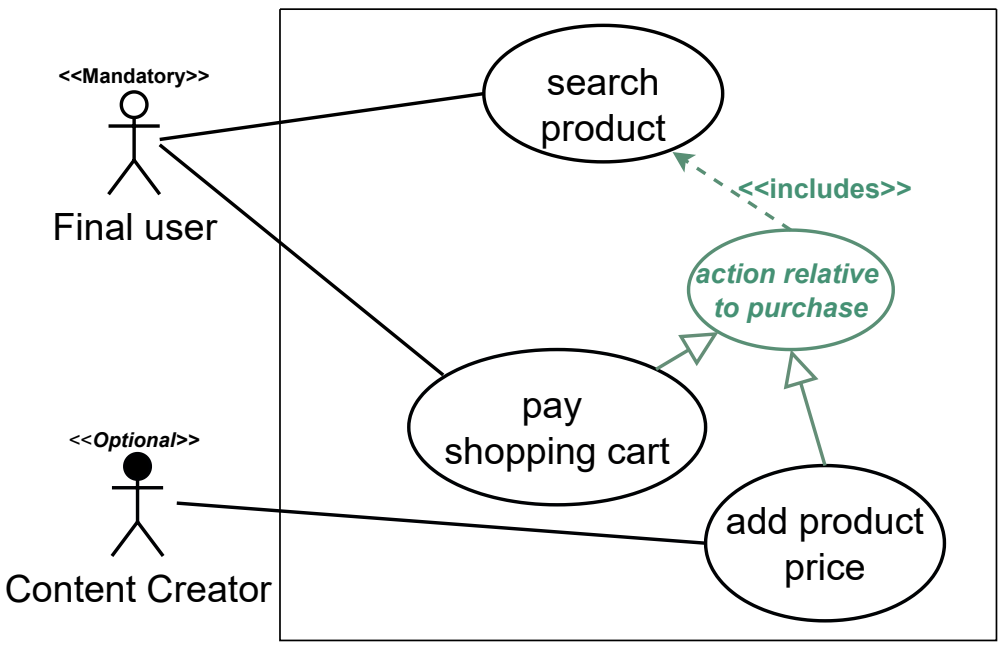
## Understandable implications

- Currently for users familiar with TCA
- More complex to interpret than dyadic implications
- Example:
  - Between features
  - $f1 \rightarrow f2$ states that when there is a triple $(s, r, f1)$, then there is a triple $(s, r, f2)$.
  - E. g. $browse\_productlist,\ update\_userprofile \Rightarrow search\_product$
  - Means: when a role can browse a product list and update a user profile in a system, they can also search products
  - And not exactly: when a system has the features 'browse a product list' and 'update a user profile', it has feature 'search products'

# Case Study

## Informative implications

- Can be used to establish a graphical variability model + additional textual constraints
  E.g. inspired from the dataset
  $(Content\ creator; add\_product\_price)$,
  $(Final\ user; pay\_shoppingcart) \rightarrow$
  $(Final\ user; search\_product)$



# Conclusion

## Overall...

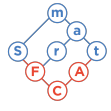
- The number of implications is surprisingly okay
- Triadic implications are too difficult to interpret
- The rules are truly informative

## Research Agenda

- Addressing more complex features (e.g. with attributes)
- Making implications efficient for Software Engineers:
  - A reading guide likely needed for the raw form
  - Translation rules towards (graphical) variability model

# Thank you for your attention !

https://dx.doi.org/10.1016/j.ijar.2024.109248




ANR SmartFCA project
Grant ANR-21-CE23-0023 anr


Junior, E. A. O., de Souza Gimenes, I. M., and Maldonado, J. C. (2010).
Systematic management of variability in uml-based software product lines.
J. Univers. Comput. Sci., 16(17):2374–2393.


Kang, K.-C. (1990).
Feature-oriented Domain Analysis (FODA): Feasibility Study; Technical Report CMU/SEI-90-TR-21 - ESD-90-TR-222.
Software Engineering Inst., Carnegie Mellon Univ.