Comparison of FP-Growth and ID3 by Creating Feature Model for Software Product Line

Win Pa Pa Htun
University of Computer Studies,
Mandalay (UCSM)
Mandalay, Myanamar

Khin Mar Myo
University of Computer Studies,
Mandalay (UCSM)
Mandalay, Myanmar

***Abstract*—Today, software product line engineering (SPLE) is an emerging software engineering paradigm. SPLE is based on the concept of reusing software artifacts gaining from the previous software product lifecycle. For SPLE, many researchers concerns with domain analysis and feature modeling (FM). In this system, the feature model management process is proposed for SPLE about mobile phone environment. To create feature model, there are two processes. The first is feature extraction process that is extracted feature from the tested mobile environment. For feature extraction process, this system uses iterative dichotomiser-3 (ID-3) and frequent pattern growth (FP-growth) algorithms. Moreover, this system compares these two feature extraction algorithms to know which algorithm is more effective for feature extraction process. By using extracted features, this system uses the propositional formula to create feature model for SPLE. This system is tested by using different HTC mobile versions. According to the performance comparison results, FP-growth is faster than ID-3 about feature extraction process.**

***Keywords*—SPLE, FP-Growth, ID-3, HTC Mobile Feature.**

I. INTRODUCTION

In the software engineering, software product line engineering (SPLE) is a process that delivers reusable components, which can be reused to develop a new application for the domain. SPLE defines software product lines requirements, software architecture and a set of reusable components, shared by the products, which implements a considerable part of the products' functionalities.

The purpose of SPLE is to reduce the time and costs of production and to increase the software quality by reusing elements which have been already tested and secured. These objectives can be realized by putting in common development artifacts such as requirement's documents, conception's diagrams, architectures, codes, test's procedures, and maintenance's procedures. The general process of product lines is based on the reusability of requirements, architecture and components.

For the software product line, the proposed system is implemented as the performance comparison system for feature model management. For performance comparison, this system compares the feature extraction process of FP-Growth and ID-3 algorithms. By using HTC extracted mobile features, this system uses propositional formula to product HTC feature model.

The main objective of the proposed system is to extract the product features from high-level software artifacts rather than low-level artifacts. This system also proposes a feature diagram by means of data mining method that includes FP-growth and ID-3. And then, this system also describes the performance compassion results. Finally, this system finds out common & variability analysis among the different version of same or different HTC software product.

This paper is organized into six sections. In the second section, related works are described. System design is shown in third section. Proposed methodology is described in fourth section. Implementation and experimental results are described in section five and six. Finally, conclusion is described in the seventh section.

II. RELATED WORK

In 2013, B. Zhang [1] presented the system about mining complex feature correlations from large software product line configurations. The input of our approach is product configurations separately documented in each product. The first process is configuration extraction which analyzes all existing product configurations and results into a configuration matrix consisting of selected features with their values across all products. Then the second process of data preparation adapts the information in the configuration matrix by unifying the data format and discretizing continuous feature values.

In 2014, B. Danilo and D. Mark [2] expressed the software product line engineering (SPLE) with feature models. One increasing trend in software development is the need to develop multiple, similar software products instead of just a single individual product. There are several reasons for this. Products that are being developed for the international market must be adapted for different legal or cultural environments, as well as for different languages, and so must provide adapted user interfaces. Because of cost and time constraints it is not possible for software developers to develop a new product from scratch for each new customer, and so software reuse must be increased.

III. PROPOSED SYSTEM DESIGN

For SPLE, this system is proposed as the performance comparison system using FP-Growth and ID-3 algorithms. Firstly, this system extracts web pages about HTC mobile phone. And then, the user must choose one of two methods that are FP-Growth and ID-3 algorithms. In this system, these algorithms are used to extract HTC mobile features and HTC feature diagram.

After extracting feature diagram by using each algorithm, this system compares the performance of each algorithm about feature extraction process. Proposed system design is shown in Figure 1.

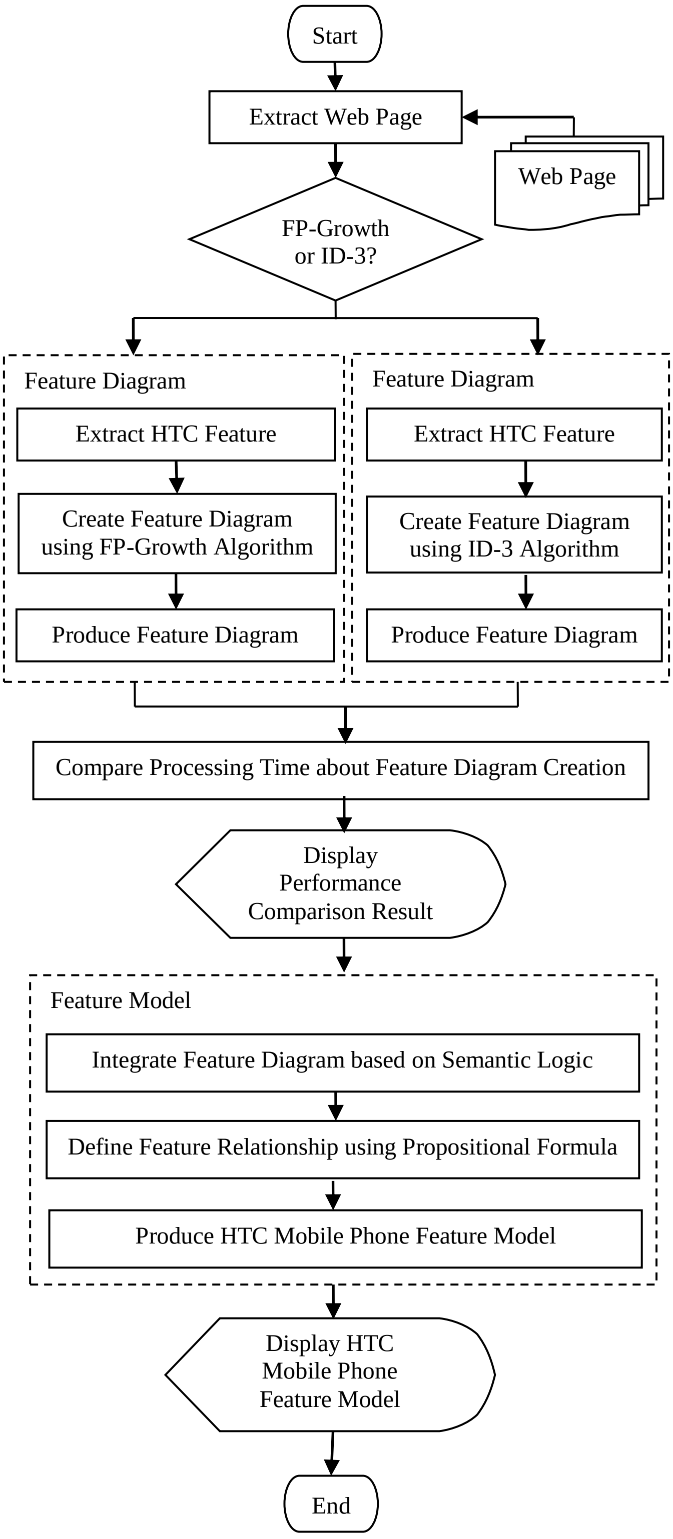

Start
Extract Web Page
Web Page
FP-Growth or ID-3?
Feature Diagram
Extract HTC Feature
Create Feature Diagram using FP-Growth Algorithm
Produce Feature Diagram
Feature Diagram
Extract HTC Feature
Create Feature Diagram using ID-3 Algorithm
Produce Feature Diagram
Compare Processing Time about Feature Diagram Creation
Display Performance Comparison Result
Feature Model
Integrate Feature Diagram based on Semantic Logic
Define Feature Relationship using Propositional Formula
Produce HTC Mobile Phone Feature Model
Display HTC Mobile Phone Feature Model
End

Fig 1. Proposed System Design

By using extracted feature diagram, this system creates HTC feature model using propositional formula. To create feature model (FM), this system first integrates feature diagram based on semantic logic. And then, this system defines each feature relationships based on propositional formula. Finally, this system produces feature model about HTC mobile phone.

IV. PROPOSED METHODOLOGY

In this section, tow data mining methods are described for feature extraction process. These methods are FP-Growth and ID-3 algorithms. And then, the propositional formula is used for feature model creation process.

A. *Frequent Pattern Growth (FP-Growth)*

Frequent pattern growth (FP-Growth) approach is an efficient approach for producing the frequent itemsets without generation of candidate itemsets. It is based upon the divide and conquers strategy [3]. Instead of generating a large number of candidates, the FP-growth approach preserves the essential groupings of the original data elements for mining. Then the analysis is focused on counting the frequency of the relevant data sets instead of candidate sets. Instead of scanning the entire database to match against the whole corresponding set of candidates in each pass, the method partitions the data set to be examined as well as the set of patterns to be examined by database projection. Such a divide-and-conquer methodology substantially reduces the search space and leads to high performance.

With the growing capacity of main memory and the substantial reduction of database size by database projection as well as the space needed for manipulating large sets of candidates, a substantial portion of data can be put into main memory for mining. New data structures and methods, such as FP-tree and pseudo-projection, have been developed for data compression and pointer-based traversal.

FP-growth may eliminate or substantially reduce the number of candidate sets to be generated and also reduce the size of the database to be iteratively examined, and therefore, lead to high performance [4]. The FP-growth approach consists of two steps:

- Constructing an FP-tree: The first step constructs a compact data structure called FP-tree that efficiently stores frequent patterns of a transaction database and enables efficient frequent pattern mining.
- Mining patterns using an FP-tree: The second step uses an FP-tree to recursively mine all frequent patterns [4].

1) FP-Growth Algorithm: FP-growth algorithm for discovering frequent pattern without candidate generation is as follows [5]:

Algorithm: FP-Growth. Mine frequent patterns using an FP-tree by pattern fragment growth.

Input: A transaction database, D; minimum support threshold, min-sup.

Output: The complete set of frequent patterns.

Method:

1. The FP-tree is constructed in the following steps.
   (a) Scan the transaction database D once. Collect the set of frequent items F and their supports. Sort F in support descending order as L, the list of frequent items.

(b) Create the root of an FP-tree and label it as “null”. For each transaction Trans in D do the following. Select and sort the frequent items in Trans according to the order of L. Let the sorted frequent items list in Trans be [p|P], where p is the first element and P is the remaining list. Call insert-tree ([p|P], T). If T has a child N such that N.item-name = p.item-name, then increment N’s count by 1; else create a new node N, and let its count be 1, it parent link be linked to T, and its node-link to the nodes with the same item-name via the node-link structure. If P is nonempty, call insert-tree (P, N) recursively.

2. Mining of an FP-tree is performed by calling FP-growth (FP-tree, Null), which is implemented as follows:

```
Procedure FP-growth (Tree, $\alpha$)
(1) if Tree contains a single path P then
(2) for each combination (denoted as $\beta$) of the nodes
    in the path P
(3) generate pattern $\beta$ U $\alpha$ with support = minimum
    support of nodes in $\beta$;
(4) else for each $a_i$ in the header of Tree {
(5) generate pattern $\beta$ = $a_i$ U $\alpha$ with support = $a_i$.
    support;
(6) construct $\beta$ ’s conditional pattern base and then $\beta$ ’s
    conditional FP-tree $\text{Tree}_\beta$;
(7) if $\text{Tree}_\beta \neq \phi$ then
(8) call FP-growth ($\text{Tree}_\beta, \beta$); }
```

B. *Iterative Dichotomiser-3 (ID-3)*

In ID-3, a decision tree is a flow-chart-like tree structure that employs a top down. To select the test attribute at each node in the tree, the information gain measure is used. The decision tree is built by using the attribute selection measure equation and decision tree algorithm [6].

1) Decision Tree Algorithm: This algorithm is as follows:

```
Algorithm : Generate_decision_tree.
Step 1: create a node N;
Step 2: if samples are all of the same class, C then return N
        as a leaf node labeled with the class C;
Step 3: if attribute-list is empty then return N as a leaf node
        labeled with the most common class in samples;
Step 4: select test-attribute, the attribute among attribute-list
        with the highest information gain;
Step 5: label node N with test-attribute;
Step 6: for each known value $a_i$ of test-attribute
          - grow a branch from node N for the condition
            test-attribute=$a_i$;
Step 7: let $s_i$ be the set of samples in samples for which test-
        attribute=$a_i$;
Step 8: if $s_i$ is empty then attach a leaf labeled with the most
        common class in samples;
        else attach the node returned by Generate _decision
        _tree;
```

2) Attribution Selection Measure: Information gain measure is used to select the test feature at each node in the tree. The feature with the highest information gain is chosen as the test attribute for the current node. Let S be a set consisting of *s* data samples. Suppose the class label attribute has m distinct values defining m distinct classes, C_i (for i=1,..,m). Let si be the number of samples of S in class C_i. The expected information needed to classify a given sample is given by

$$I(s_1,s_2,\ldots,s_m) = -\sum_{i=1}^{m} p_i \log(p_i) \qquad (1)$$

where p_i is the probability that an arbitrary sample belongs to C_i and is estimated bys_i/s.

Let attribute A have v distinct values, {a_1, a_2,…, a_v}. Attribute A can be used to partition S into v subsets, {S_1, S_2,…, S_v}, where S_j contains those samples in S that have value a_j of A. Let s_{ij} be the number of samples of class C_i in a subset S_j. The entropy, or expected information based on the partitioning into subsets by A, is given by

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j},\ldots,s_{mj}}{s} I(s_{1j},\ldots,s_{mj}) \qquad (2)$$

The term $\frac{s_{1j},\ldots,s_{mj}}{s}$ acts as the weight of the jth subset and is the number of samples in the subset divided by the total number of samples in S. For a given subset S_i,

$$I(s_{1j},s_{2j},\ldots,s_{mj}) = -\sum_{i=1}^{m} p_{ij} \log(p_{ij}) \qquad (3)$$

where $p_{ij} = s_{ij}/|S_j|$ and is the probability that a sample in S_j belongs to class C_i.

The encoding information that would be gained by branching on A is

$$\text{Gain}(A) = I(s_1,s_2,\ldots,s_m) - E(A) \qquad (4)$$

The feature with the highest information gain is chosen as the test feature for the given set S. A node is created and labeled with the feature, branches are created for each value of the feature, and the samples are partitioned accordingly [6].

C. *Propositonal Formula*

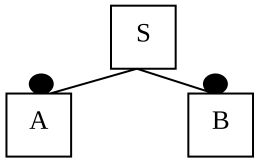
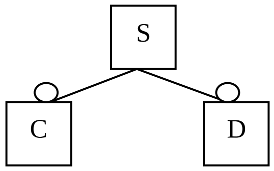
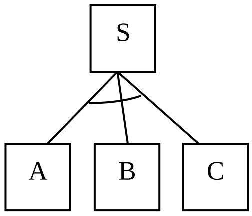
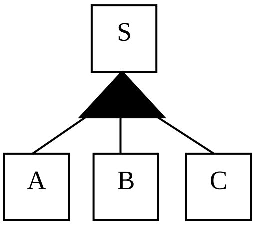
To create the feature model, this system uses the propositional formula. The semantics of a feature model is the set of feature configurations that the feature model permits. The most common approach is to use mathematical logic to capture the semantics of a feature diagram. Each feature corresponds to a boolean variable and the semantics is captured as a propositional formula. The satisfying valuations of this formula correspond to the feature configurations permitted by the feature diagram. For instance, if f_1 is a mandatory sub-feature of f_2, the formula will contain the constraint $f_1 \Leftrightarrow f_2$.

Relationships between a parent feature and its child features (or sub-features) are categorized as mandatory (child feature is required), optional (child feature is optional), Or (at least one of the sub-features must be selected) and Alternative (xor) (one of the sub-features must be selected).

In addition to the parental relationships between features, cross-tree constraints are allowed. The most common are: A requires B (The selection of A in a product implies the selection of B) and A excludes B (A and B cannot be part of the same product) [7, 8].

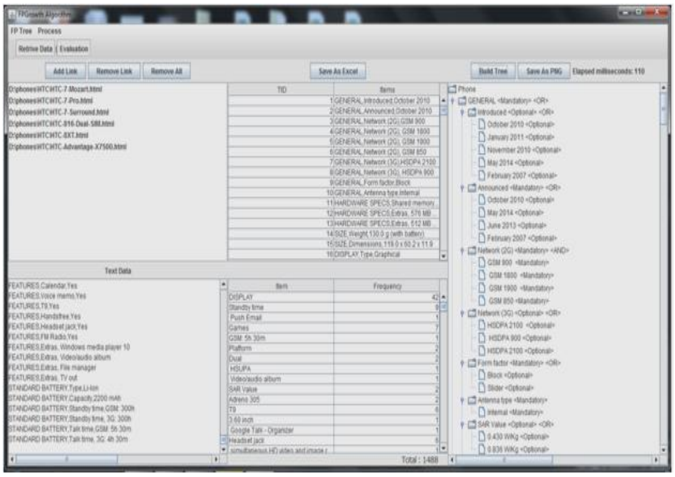
Propositional formula is shown in Table I.

TABLE I. PROPOSITIONAL FORMULA

Concept	Diagram	Propositional Formula
Mandatory		$S \leftrightarrow B \quad \wedge \quad S \leftrightarrow A$
Optional		$C \leftrightarrow S$ $D \leftrightarrow S$
OR		$(S \leftrightarrow A \quad \vee B \vee C) \quad \wedge$ almost1 (A, B, C)
X-OR		$S \leftrightarrow A \quad \vee B \vee C$

V. IMPLEMENTATION OF THE PROPOSED SYSTEM

This system is implemented by using Java programming language. Feature model using FP-growth is shown in Figure 2 and feature model using ID-3 is shown in Figure 3.



Elapsed milliseconds: 110

D:\phones\HTC\HTC-7-Mozart.html
D:\phones\HTC\HTC-7-Pro.html
D:\phones\HTC\HTC-7-Surround.html
D:\phones\HTC\HTC-816-Dual-SIM.html
D:\phones\HTC\HTC-8XT.html
D:\phones\HTC\HTC-Advantage-X7500.html

TID	Items
1	GENERAL Introduced October 2010
2	GENERAL Announced October 2010
3	GENERAL Network (2G) GSM 900
4	GENERAL Network (2G) GSM 1800
5	GENERAL Network (2G) GSM 1900
6	GENERAL Network (2G) GSM 850
7	GENERAL Network (3G) HSDPA 2100
8	GENERAL Network (3G) HSDPA 900
9	GENERAL Form factor Block
10	GENERAL Antenna type Internal
11	HARDWARE SPECS Shared memory
12	HARDWARE SPECS Extras, 576 MB
13	HARDWARE SPECS Extras, 512 MB
14	SIZE Weight 130.0 g (with battery)
15	SIZE Dimensions 119.0 x 60.2 x 11.9
16	DISPLAY Type Graphical

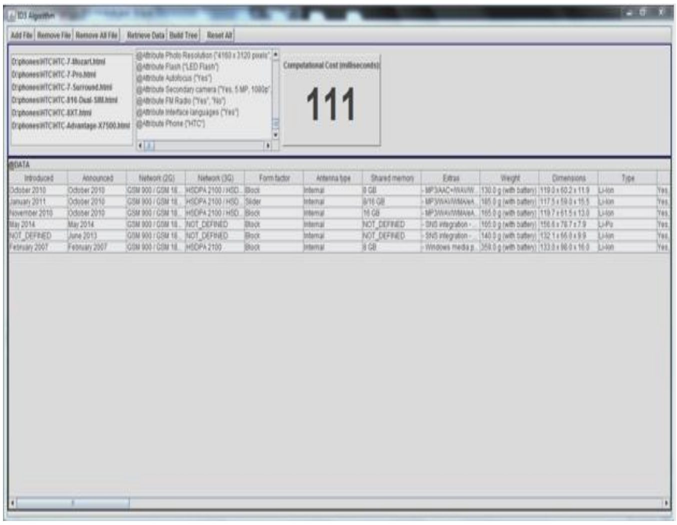
FEATURES.Calendar.Yes
FEATURES.Voice memo.Yes
FEATURES.T9.Yes
FEATURES.Handsfree.Yes
FEATURES.Headset jack.Yes
FEATURES.FM Radio.Yes
FEATURES.Extras. Windows media player 10
FEATURES.Extras. Video/audio album
FEATURES.Extras. File manager
FEATURES.Extras. TV out
STANDARD BATTERY.Type.Li-Ion
STANDARD BATTERY.Capacity.2200 mAh
STANDARD BATTERY.Standby time.GSM: 300h
STANDARD BATTERY.Standby time, 3G: 300h
STANDARD BATTERY.Talk time.GSM: 5h 30m
STANDARD BATTERY.Talk time, 3G: 4h 30m

Item	Frequency
DISPLAY	42
Standby time	9
Push Email	1
Games	7
GSM: 5h 30m	1
Platform	2
Dual	2
HSUPA	1
Video/audio album	1
SAR Value	2
Adreno 305	2
T9	6
3.60 inch	1
Google Talk - Organizer	1
Headset jack	6
simultaneous HD video and image r	1

Total : 1488

- Phone
  - GENERAL <Mandatory> <OR>
    - Introduced <Optional> <OR>
      - October 2010 <Optional>
      - January 2011 <Optional>
      - November 2010 <Optional>
      - May 2014 <Optional>
      - February 2007 <Optional>
    - Announced <Mandatory> <OR>
      - October 2010 <Optional>
      - May 2014 <Optional>
      - June 2013 <Optional>
      - February 2007 <Optional>
    - Network (2G) <Mandatory> <AND>
      - GSM 900 <Mandatory>
      - GSM 1800 <Mandatory>
      - GSM 1900 <Mandatory>
      - GSM 850 <Mandatory>
    - Network (3G) <Optional> <OR>
      - HSDPA 2100 <Optional>
      - HSDPA 900 <Optional>
      - HSDPA 2100 <Optional>
    - Form factor <Mandatory> <OR>
      - Block <Optional>
      - Slider <Optional>
    - Antenna type <Mandatory>
      - Internal <Mandatory>
    - SAR Value <Optional> <OR>
      - 0.430 W/Kg <Optional>
      - 0.836 W/Kg <Optional>

Fig 2. Feature Model using FP-Growth



D:\phones\HTC\HTC-7-Mozart.html
D:\phones\HTC\HTC-7-Pro.html
D:\phones\HTC\HTC-7-Surround.html
D:\phones\HTC\HTC-816-Dual-SIM.html
D:\phones\HTC\HTC-8XT.html
D:\phones\HTC\HTC-Advantage-X7500.html

@Attribute Photo Resolution ("4160 x 3120 pixels"
@Attribute Flash ("LED Flash")
@Attribute Autofocus ("Yes")
@Attribute Secondary camera ("Yes, 5 MP, 1080p"
@Attribute FM Radio ("Yes", "No")
@Attribute Interface languages ("Yes")
@Attribute Phone ("HTC")

Computational Cost (milliseconds)

111

@DATA

Introduced	Announced	Network (2G)	Network (3G)	Form factor	Antenna type	Shared memory	Extras	Weight	Dimensions	Type	
October 2010	October 2010	GSM 900 / GSM 18...	HSDPA 2100 / HSD...	Block	Internal	8 GB	- MP3/AAC+/WAV/W...	130.0 g (with battery)	119.0 x 60.2 x 11.9	Li-Ion	Yes,
January 2011	October 2010	GSM 900 / GSM 18...	HSDPA 2100 / HSD...	Slider	Internal	8/16 GB	- MP3/WAV/WMA/eA...	185.0 g (with battery)	117.5 x 59.0 x 15.5	Li-Ion	Yes,
November 2010	October 2010	GSM 900 / GSM 18...	HSDPA 2100 / HSD...	Block	Internal	16 GB	- MP3/WAV/WMA/eA...	165.0 g (with battery)	119.7 x 61.5 x 13.0	Li-Ion	Yes,
May 2014	May 2014	GSM 900 / GSM 18...	NOT_DEFINED	Block	Internal	NOT_DEFINED	- SNS integration - ...	165.0 g (with battery)	156.6 x 78.7 x 7.9	Li-Po	Yes,
NOT_DEFINED	June 2013	GSM 900 / GSM 18...	NOT_DEFINED	Block	Internal	NOT_DEFINED	- SNS integration - ...	140.0 g (with battery)	132.1 x 66.0 x 9.9	Li-Ion	Yes,
February 2007	February 2007	GSM 900 / GSM 18...	HSDPA 2100	Block	Internal	8 GB	- Windows media p...	359.0 g (with battery)	133.0 x 98.0 x 16.0	Li-Ion	Yes,

Fig 3. Feature Model using ID-3 Algorithm

VI. EXPERIMENTAL RESULTS OF THE PROPOSED SYSTEM

This system is tested by using different versions of HTC mobile phone. According to the experimental results, the feature extraction processing time of FP-growth is faster than the processing time of ID-3. So, the FP-growth algorithm is more effective than ID-3 algorithm for software product line engineering (SPLE). Some experimental results are shown in Table 2.

TABLE II. EXPERIMENTAL RESULTS

ID	Version Name	Processing Time of FP-Growth	Processing Time of ID-3
1	HTC-7-Mozart (V1) and HTC-7-Pro (V2)	75	372
2	HTC-7-Mozart (V1), HTC-7-Pro (V2) and HTC-7-Surround (V3)	72	90
3	HTC-7-Mozart (V1), HTC-7-Pro (V2), HTC-7-Surround (V3) and HTC-8XT (V4)	97	117
4	HTC-7-Mozart (V1), HTC-7-Pro (V2), HTC-7-Surround (V3), HTC-816-Dual-SIM (V4), HTC-8XT (V5) and HTC-Advantage-X7500 (V6)	110	111

Performance analysis results are shown in Figure 4.

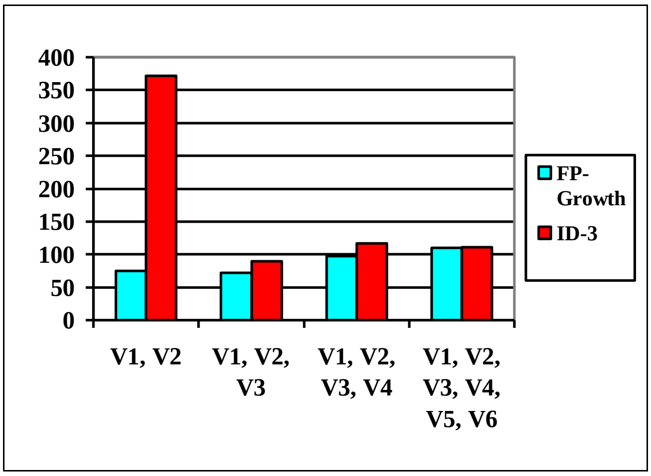


Fig 4. Performance Analysis Result

VII. CONCLUSION

In conclusion, the proposed system provides the software product line by producing the feature model about mobile phone. This system is tested by using many web pages that contain HTC mobile phone information. By using the proposed system, the software developer can estimate the features about the coming mobile phone product. Moreover, this system compares the performance about feature extraction process. So, this system allows the user to know which data mining method is more effective about feature extraction process. So, the proposed system provides many benefits for the software product line about the mobile phone domain.

REFERENCES

[1] B. Zhang, "Mining Complex Feature Correlations from Large Software Product Line Configurations", Software Engineering Research Group, University of Kaiserslautern, Germany, Technical Report of AGSE, April 3, 2013.

[2] B. Danilo Beuche and D. Mark, “Software Development Magazine – Programming”, Software Testing, Project Management, Jobs, 2014.

[3] G. Kaur and S. Aggarwal, "Performance Analysis of Association Rule Mining Algorithms", International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, August, 2013, pp. 856-858.

[4] J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation”, In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, USA.

[5] J. Han and M. Kamber, "Data Mining: Concepts and Techniques", Simon Fraser University, United States of America, 2001.

[6] H. Jiawei and K. Micheline, "Data Mining Concepts and Techniques", Simon Fraser University, United States of America, 2001.

[7] K. Czarnecki and S. Helsen and U. Eisenecker, "Staged configuration using feature models", Proceedings of the Third International Conference on Software Product Lines (SPLC '04), vol 3154, Springer, August 2004.

[8] K. Czarnecki and A. Wasowski, "Feature Diagrams and Logics: There and Back Again", University of Waterloo, Canada, 2008.