Analytical Product Release Planning

Maleknaz Nayebi, Guenther Ruhe

Software Engineering Decision Support Laboratory, University of Calgary
2500 University Drive NW, Calgary, AB, T2N 1N4 Canada

ABSTRACT

As part of any incremental and iterative development, release planning is the process of assigning features to upcoming releases (or iterations) such that the overall product evolution is optimized. Analytical product release planning refers to the application of analytical methods in this process, thereby utilizing the diversity of data available from internal and external sources of information. In this chapter, information needs for release planning are outlined and a taxonomy of release planning problems is given. The paradigm of Open Innovation is introduced as a new way to elicit and get access to relevant data related to product objectives, features and their dependencies, customers and changing priorities, as well as product values and market trends. Analytical Open Innovation (AOI) is the integration of Open Innovation with (a portfolio of) analytical methods which could be used in different problems of semi-wicked nature such as planning and design. This chapter studies the usage of AOI in the context of release planning (RP). The respective approach called AOI@RP is taking advantage of gathering and generating data and relating data into well-defined aspects of the problem and combining analytical methods to address the solution. The usage of AOI is studied in more detail for two of the concrete release planning problems given in the taxonomy: (i) Release planning in the presence of advanced feature dependencies and synergies detected from morphological analysis, (ii) Continuous what-to-release planning in consideration of ongoing trial feature evaluation. An illustrative case study is used as the proof of concept to the proposed solution methodology.

Keywords: Release planning, open innovation, data analytics, decision support, case study.

1. Introduction and motivation

Release planning is a decision-centric problem with comprehensive information and knowledge needs. A release is a major (new or upgraded) version of an evolving product characterized by a collection of (new, corrected or modified) features. Good release planning is a mandatory part of incremental and iterative software development. Release decisions address the questions about the functionality (what?), time (when?) and quality (how good?) of offering product releases. All types of release decisions are part of software product management, the discipline of governing a software product from its inception to the market until it is closed down [1].

Adaptive software development [2] as well as other domain of product development is increasingly affected by ongoing changes in business conditions. It also raises the need to adapt related decision-making processes. The formerly reactive mode of operation needs to be replaced by real-time or pro-active decisions based on highly up to date and comprehensive information. Big data analytics offers the principal pathway to make pro-active decisions. Information about changing customer and stakeholder product preferences and demands, knowledge about possible feature synergies in terms of added value or reduced effort estimation, as well as the possibility of performing pro-active product evaluation will increase the likelihood of developing the "right" product. A product manager can get additional information about product performance and mine industry trends and investigate on customer needs for achieving actionable insight.

The paradigm of Open Innovations is emphasizing on the range of opportunities available to get access to distributed knowledge and information. Open innovation integrates internal and external ideas and paths to market at the same level of importance [3] and merges distributed talent, knowledge and ideas into innovation processes [4]. Analytical Open Innovation (AOI) approach is designed in this context in order to make use of more knowledge containers for comprehensive decision making and to address wickedness of the problem under study. We define AOI as being the integration of Open Innovation with (a portfolio of) analytical methods. AOI is positioned as a response to existing challenges of mining software repositories in software engineering. Some the challenges are listed as follow:

- Mainly tame (well defined) problems are considered. However, many software engineering planning and design problems incorporate some form of wickedness [5-7].
- Primarily, internal knowledge and repositories are used for the mining process. However, knowledge is available and retrievable from a broader range of information sources [5, 6].
- Often, a "closed world" and quantitative type of analysis is considered. However, problems need qualitative analysis as well as human expertise [8, 9].
- Mining software repositories (MSR) efforts, are mainly intended to support developers and increasingly project managers, but the role of a product manager is largely ignored so far [7, 10, 11].

Current release planning methods are largely based on a "Closed Innovation", with information, knowledge and stakeholders introduced (just) being largely static and pre-defined. Main deficits of current release planning techniques are related to their inability to handle the large amounts of data related to the ongoing change that is happening in the underlying development and business processes. In this chapter (i) we propose the analytical Open Innovation (AOI) also (ii) we discuss the application of AOI to the area of release planning decision support and will call this AOI@RP.

In this book chapter, a taxonomy of (data-intensive) release planning problems is given in Section x.2 Therein, special emphasis is on the content and impact of data analytics in all these problems. Information needs for software release planning are studied in Section x.3. Data analytics techniques, as part of open innovation, are discussed in more detail in Section x.4. The techniques are applied for an illustrative case study presented in Section x.5. Finally, an outlook is presented on future research in Section x.6.

2. Taxonomy of Data-intensive Release Planning Problems

There is a wide range of decision problems that altogether are subsumed under the term "Release planning". In what follows, seven classes of release planning problems are described.

2.1 What-to-release Planning

As part of any incremental and iterative software development, the question which new features should be offered in upcoming releases is of key importance for product success. From an existing pool of features, the selection and schedule decisions are highly information intensive. To decide about the features, and assign them to one of the upcoming releases, one requires a good understanding of the features, their market value, their mutual dependencies and synergies, their effort to get them implemented and the market needs and trends to select the most appropriate feature for enhancing or updating the existing product. The main difficulty of the problem is that a large portion of the requested information is continuously changing.

2.2 Theme-based Release Planning

The question of what constitutes a good release (content) is hard to answer. The formulation given in Section x.4.2.1 assumes that the total value of a release is the sum of the value of the individual features, implemented and offered in this release. However, this is likely just an approximation of the truth. Some features are highly dependent and would have higher value when they are released along with a specific set of features.

A theme is meta-functionality of a product release, integrating a number of individual features under a joint umbrella. It can be thought of as an abstraction, i.e., a group of features that are inter-related to each other in a way that they depict a context and can be viewed as a single entity from a higher level [12, 13].

Theme based release planning not only considers the value of individual features, but also utilizes synergy effects between semantically related features. Detection and formulation of dependencies are supported by performing cross consistency assessment (CCA) as discussed in Section X.4.2.2. Pairwise consistency assessment between features shows the explicit dependencies and synergies, as well as hidden dependencies and synergies between features.

2.3 When-to-release Problem

When-to-release decisions are trade-off decisions by their nature. The product manager needs to balance the possible gain (expressed by the value function) against the potential risks of delivering (too) early. This risk here is essentially related to quality and customer acceptance. Although it is difficult to express formally, our assumption is that the risk is higher if the earlier the release date is chosen.

Various factors are relevant for deciding about when-to-release. The notion of *release readiness* incorporates a variety of factors related to requirements, coding and testing. Port et al. [14] illustrated the value of knowing RR in project success with aid of an explorative study at NASA's Jet Propulsion Laboratory (JPL). RR facilitates confident release decisions along with proactive addressing of problems related to releases. A list of frequently used release attributes and factor is given in Table x.2

Table 1. List of release readiness attributes and related metrics

Attributes (C_i)	RR Metrics (M_i)
Satisfaction of feature completion	Feature Completion Rate (FCR)
Satisfaction of features implemented	Features Implemented (FI)
Satisfaction of build/continuous integration trends	Build Success Rate (BSR)
Satisfaction of implementation effort	Code Churn Rate (addition and deletion per day) (CCR)
Satisfaction of defect finding	Defect Find Rate (DFR)
Satisfaction of defect fixing	Bug Fixing Rate (BFR)
Satisfaction of change completion	Change completion Rate (CR)
Satisfaction of pull request completion	Pull-request Completion Rate (PCR)

2.4 Release Planning in Consideration of Quality

The traditional view of release planning favors delivery of pieces of functionality (called features or requirements, depending on the granularity) in a best possible way. For a feature to become part of a product release it needs to be implemented and thus, it consumes resources. The core question at this point is how to utilize the resources in the best possible way to provide the best combination of pieces of functionality. However, what is completely lacking in this view is the quality aspect of the resulting release product(s).

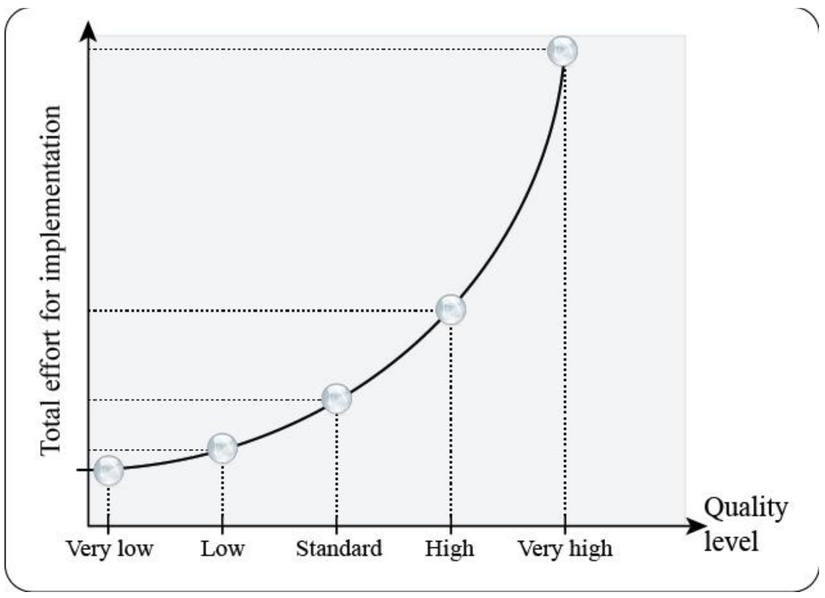


Figure 1. Total implementation effort of a feature in dependence of the target quality level

Depending on the granularity of the planning, the specific effort attributed to higher target levels of quality can be feature related and/or related to the whole product release (as a form of cross-cutting concern). The relation between quality and effort is demonstrated in Figure 1. With this formulation given, the problem is no longer just finding a plan to implement the most comprehensive and attractive set of features. Instead, the problem now becomes a trade-off analysis where the balance is between providing the most comprehensive and attractive functionality in dependence of varying levels of target quality. A prototype tool for supporting release planning of quality requirements and its initial industrial evaluation has been presented by Svensson et al. [15]. As part of the underlying planning method called QUPER [16], the tool helps to reach an alignment between practitioners, e.g., product managers and developers, of what level of quality is actually needed [12].

2.5 Operational Release Planning

Once a strategic release plan is confirmed, the question becomes how to realize this plan. At this stage, planning gets more detailed, taking into account the existing pool of developers, their skills, as well as the order of tasks needed to implement the features. The problem; called *Operational Release Planning;* is to assign developers in the best possible way to the tasks constituting the features. It includes scheduling of tasks in consideration of possible technological constraints. Typically, the time horizon for operational planning is just before the upcoming release (or iteration).

For the staffing problem, each feature is considered as the result of performing a sequence of tasks. All objectives and constraints of the staffing problem(s) are formulated in terms of features and their related tasks. The solution method to the staffing problem considers the characteristics of the individual tasks and makes suggestions about the allocation of developers to the tasks and the order in which they should be performed. Among the tasks related to the same feature, there might be different types of dependencies. Another conceptual component looks into requested competences for performing the tasks [12].

Project scheduling as part of software project management was studied, for example, by Alba and Chicano [17]. In order to decide which resources are used to perform which tasks and when each task should be performed, the authors suggested applying genetic algorithms (GA's).

2.6 Release Planning in Consideration of Technical Debt

The metaphor of technical debt [18] refers to the actual versus ideal status of an evolving software system. Analogously to financial debt, different types of opportunistic development shortcuts and related temporary advantages (debt) are accepted in terms of the systems design, coding, testing and documentation. In organizations, it is very common to strategically put off fixing a (non-critical) issue in order to implement new features as proposed by the release planning process.

It is a challenge for organizations to manage and track technical debt [19]. Accumulated debts, without appropriate refactoring, could threaten the maintainability of the products, and hinder future development due to unpredictable and poorly designed architecture. Mining data can help in understanding the amount and potentially the root causes of the technical debt and what can be done about that principle (refactoring, redesign, etc.). Technical debt needs to be measured and tracked since the requirements gathering stage. Software design and requirements engineering need to identify the right, meaningful things to measure, such as the number of assumptions you make, the number of postponed functionality, the decision of technology platforms and etc. A wrong requirement is already a debt on future development as the team will need to redesign or rebuild already made items to fit into requirements.

2.7 Release Planning for Multiple Products

Instead of looking into the planning of just one product, often a suite of related products is considered. This increases not only the functionality, but also allows better customization, providing the customer with options to select the most appropriate products of the suite. As the development of a product suite typically follows the same paradigm of incremental and evolutionary development, the problem of release planning for one product is generalized that way to a problem with multiple products. At this point, the product can be either software or hardware related or a mixture of both.

The higher complexity of planning for a product suite instead of just a single product results from the demand that the individual products of the suite should be offered synchronously. If this is not the case, a substantial part of the overall inherent value expected from offering a product suite is jeopardized, and/or the release of the suite needs to be delayed until the last part in the chain is finished as well.

Planning for release of product lines is another emerging topic. Schubanz et al. [20] proposed a conceptual model and corresponding tool support to plan and manage the systematic evolution of software-intensive systems. Their approach allows providing support for in time, continuous planning over long periods of time and many releases.

3. Information Needs for Software Release Planning

The paradigm of Open Innovation is aimed to make better knowledge and information to qualify products decision-making. Information needs in software development and management was studied by Buse and Zimmermann [5]. In this chapter, we discuss expected information needs in the context of release planning. The evaluation is based on a combination of literature research and practical experience [21-26]. In order to utilize Open Innovation techniques in the context of release planning the information needs are investigated in this section.

3.1 Features

Wiegers [18] has defined a product feature as a set of logically related requirements that provide a capability to the user and enable the satisfaction of business objectives. For the actual description of features, different types of information need to be collected in a feature repository. As an example for a description scheme, we present the structure suggested in adaptation of the scheme suggested by Regnell and Brinkkemper [27]. While all information is of strong relevance for release decisions to be made, the information is hard to retrieve and to maintain. The situation gets even worse from the degree of change inherent in the information.

Table 2. Feature characterization scheme (based upon [27])

Attribute	**Value**
State	Candidate / approved / specified / discarded / planned / developed / verified / released
ID	Unique identity
Submitter	Who issued it?
Company	Submitter’s company
Domain	Functional domain
Description	Short textual description
Contract	Link to sales contract enforcing requirement

Attribute	**Value**
Priorities	Prioritization for different criteria by different stakeholders on a nine point scale
Motivation	Rational: Why it is important?
Line of business	Market segment for which feature is important
Specification	Link to use cases, textual specification
Dependencies	Precedence, coupling or other dependencies between features
Resource estimation	Estimated effort per defined resource type
Risk	Projected risks of implementation and market penetration
Schedule	Release for which it is planned for
Design	Link to design documents
Test	Link to test documents
Novelty	Novelty of the features when compared to competitors
Release version	Official release name

3.2 Feature value

The question of what constitutes the value of a feature is difficult to answer. Value definition is context specific and user specific. The feature value is time-dependent, as the value might change under changing market or business conditions. Furthermore, the individual value of a feature is not additive towards the overall release value. Offering certain features in conjunction, e.g., related to themes (see Section 2.2), will provide synergies that are important to be taken into account [13].

A comprehensive value-map is suggested by Khurum et al. [28]. Utilizing knowledge from state-of-the-art in software engineering, business, management, and economics, and gathered through extensive literature reviews as well as working with professionals in industry, the authors studied a broad range of value constructs and classified them belong to customer, internal business, financial, as well as innovation and learning perspective. The authors also performed some industrial evaluation of the proposed taxonomy. They report a fundamental impact within Ericsson, attributed to the creation and use of the software value map, indicated by the shift from cost-based discussions and reasoning, to value-based decision support.

3.3 Feature dependencies

From analyzing industrial software product release planning projects in the domain of telecommunications, Carlshamre et al. [29] observed that features are often dependent on each other. For the projects and the domain analyzed, the percentage of features being dependent in one way or the other was as high as 80%.

There is a variety of dependencies which points into different meanings of the term “dependency”. We distinguish between dependencies in the following categories based on Dahlstedt and Persson [30]:

- Implementation space (two features are “close” in their implementation and should be treated together),
- Feature effort space (two features are impacting each other in their implementation effort if provided in the same release),
- Feature value space (two features are impacting each other in their value if provided in the same release), and
- Feature usage space (two features are only useful for the customer if provided in the same release).

Elicitation of dependencies is a form of knowledge elicitation and inherently difficult. Potentially, there are a large number of dependencies. Knowing at least the most critical ones will increase the chances to generate meaningful plans. Similarly, ignoring them will create plans lacking to fulfill required conditions between features [31].

3.4 Stakeholders

Stakeholders can play very different roles for the definition of the product planned for. They can be related to the design and/or development of the product, they can have a financial interest by the sale or purchase of the product, they can be responsible for the introduction and maintenance of the product, or they can have an interest in the use of the product. All of them are relevant, and their opinions are often contradictory. One of the challenges of product release planning is to determine a well-balanced product plan which addresses the most relevant concerns.

Sharp et al. in [32] presented an approach of constructive guidance on how to actually find “the right set” of stakeholders. As a first (baseline) approach, four groups of stakeholders are summarized:

- Users and customers,
- Developers,
- Legislators (such as professional bodies, trade unions, legal representatives, safety executives or quality assurance auditors), and
- Decision-makers (such as the CEO or shareholders).

Once the baseline stakeholders are established, the authors suggest a five-step procedure to add further stakeholders to one of the four established groups. With regard to get access to stakeholder opinion, this can originate from anywhere in the organization, or can be systemized and structured through personas.

3.5 Stakeholder opinions and priorities

Stakeholder opinions and priorities are of key relevance to the plan and design of product releases. The process refers to the determination of priorities for the features under consideration for the next release(s). With the range of different segments, stakeholder opinions need to be clustered. Sample segmentation can be done based on criteria such as

- Demographic — gender, age, race, ethnicity
- Psychographic — personality, values, attitudes, interests, lifestyles
- Behavioral — product purchase patterns and usage, brand affinity
- Geographic — location-specific by city, state, region, or country

The process of attracting stakeholder priorities is complex and reliable information is hard to get. However, without this information, product development becomes very risky. Gorschek et al. found that too often critical stakeholders are overlooked and priorities are given in an ad hoc fashion [33].

More recently, discussion forums and user groups have been used to get access to stakeholder opinions [34]. This way, partial automation of information retrieval is possible. Social network analysis is another direction to improve the requirements prioritization process. Fitsilis et al. [35] applied meta-networks where basic entities are combined for prioritizing requirements. They analyzed the required collaboration/knowledge within the requirements and analysis project team (including clients, managers, analysts, developers etc.) in order to efficiently/effectively prioritize the requirements through the identification of the proper requirements prioritization technique.

3.6 Release readiness

Release readiness is a composite attribute intended to describe the degree of readiness to ship a product release. It comprises different aspects of the processes needed to implement a software product. Key dimensions of release readiness are:

- Implementation of functionality
- Testing
- Source code quality

- Documentation

Monitoring all these dimensions with specific metrics is mandatory to further create predictive models for release readiness or analyzing the current status of it. An analytical method for evaluating release readiness is presented in [36].

3.7 Market trends

In proactive software development the analysis of customer needs is of importance and in principle the development process is different from bespoke ones [37]. A target market or target audience is the market segment which a particular product is marketed to it. It is often defined by age, gender and/or socio-economic grouping. Market targeting is the process in which intended actual markets are defined, analyzed and evaluated just before making the final decision to enter a market [38].

Information about current needs, competitors in these markets and even more so about future trends is of pivotal importance to decide about a future product release.

3.8 Resource consumptions and constraints

Development and testing of new features and their proper integration into the existing product consumes effort and includes different type of human resources. Planning without investigating the resource constraints is risky, as the resulting plans are likely to fail. However, effort prediction is known to be very difficult and impacted by many factors. Different methods are applicable for providing estimates [39]. Most of these methods incorporate some form of learning based on information available from former projects. In addition, often these methods are hybrid in the sense that they combine formal techniques with the judgment and expertise of human domain experts. In all cases, no reliable estimates can be expected without proper and up-to-date information related to factors supposed to influence effort estimates (e.g., product size, complexity, development processes, tools used, organization, productivity factors) [18].

3.9 Synthesis of Results

In this section, we gave an informal definition of seven classes of release planning problems, with special emphasis on their data and information needs. Using literature research and our practical experience [1], [21-26], we have applied information needs described in Section x.3 and provided an evaluation of the needs for the different types of release planning problems. The results are summarized in Table 2.

Table 3. Information needs for different types of release planning problem

	Information needs							
Type of release planning problem	**Features**	**Feature dependencies**	**Feature value**	**Stakeholder**	**Stakeholder opinion and priorities**	**Release readiness**	**Market trends**	**Resource consumptions and constraints**
What to release	×	×	×	×	×		×	×
Theme based	×	×	×	×	×		×	×
When to release	×	×	×	×	×	×		×
Consideration of quality requirements	×		×	×	×	×	×	×
Operational release planning	×		×					×
Consideration of technical debt	×	×				×	×	
Multiple products	×	×	×	×	×	×		×

4. The Paradigm of Analytical Open Innovation

Open innovation integrates internal and external ideas and paths to market at the same level of importance [3], and merges distributed talent, knowledge and ideas into innovation processes [4]. Analytical Open Innovation (AOI) is defined as the integration of Open Innovation with (a portfolio of) analytical methods. Its main goal is to make use of both internal and external knowledge containers for comprehensive decision making and to address wickedness of the problem under study [40]. Data analytics is the use of advanced analytical techniques against very large diverse data sets that include different types such as structured/unstructured and streaming/batch [41]. AOI subsumes all the methods, tools and techniques to extract operational knowledge and insight from existing data sets.

AOI includes a great variety of methods and techniques, such as text and data mining, reasoning and machine learning, clustering, optimization and simulation, as well as all forms of predictive models. For the approach described in this chapter, we discuss the effect of analytical techniques for leveraging release decisions in Section x.4.1. We focus on the application and integration of two analytical techniques describes in Section x.4.2. The methodology is then illustrated in Section x.5.

4.1 The AOI@RP Platform

The AOI@RP platform is intended to support the application of AOI for different classes of release planning problems. Its main architecture is shown in Figure 2. The AOI@RP platform consists of three layers. Each of them is explained in more detail in the subsequent subsections.

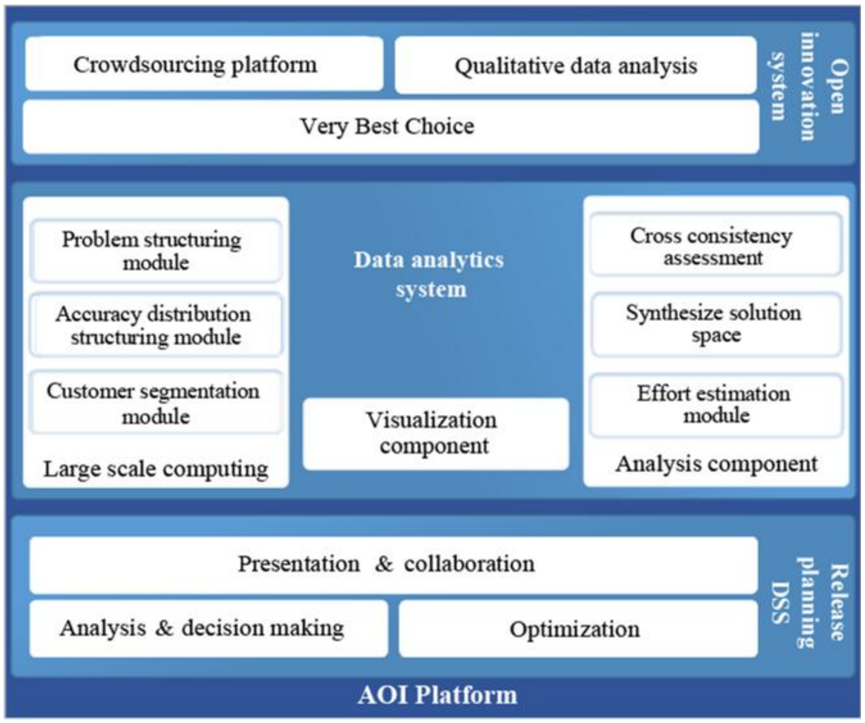

Crowdsourcing platform
Qualitative data analysis
Open innovation system
Very Best Choice
Problem structuring module
Accuracy distribution structuring module
Customer segmentation module
Large scale computing
Data analytics system
Visualization component
Cross consistency assessment
Synthesize solution space
Effort estimation module
Analysis component
Presentation & collaboration
Analysis & decision making
Optimization
Release planning DSS
AOI Platform

Figure 2. Proposed AOI platform

4.1.1 Open Innovation System

This system is designed to satisfy data needs presented in Section x.3. Data gathering and generation is covered by following the open innovation approach and crowdsourcing is selected with the aim of direct user involvement. Crowdsourcing platform provides the crowd for answering questions; facilitate controlling and verifying their works and task distribution. For now *Amazon Mechanical Turk* [42] service as a micro-task market is employed. In addition, this platform in collaboration with Very Best Choice™ maintains contact with the crowd. In order to manage collaboration between systems feedback management and representation alongside with in house collaboration with crowd, the *Very Best Choice*™ (VBC light) software is used. VBC light is a lightweight decision support system designed to facilitate proper priority decisions [43]. Text mining platform is used along with other platforms to enable automatic understanding of crowd's response in order to generate meaningful data.

4.1.2 Release Planning DSS

This layer is presented as the release planning DSS in Figure 2. The different dimensions of release planning problem is presented in Section x.2. *ReleasePlanner*™ provides a proven [12, 22, 44] functionality to facilitate voting and prioritization as well as generating optimized plans although the *ReleasePlanner*™ was designed to work in close innovation context, the underlying model of that platform needs to be adapted to the needs of handling more comprehensive data from different sources. The *Presentation & collaboration* component represents process outcomes to the organization and stakeholders. It is also responsible to initiate the work of other platforms. The *Optimization component* is responsible for computation of optimized and diversified alternative release plans based on specialized integer programming and the special structure of the problem. The *Analysis & decision component* defines alternative features and plans with their resource consumption and the degree of stakeholder excitement.

4.1.3 Data Analytics System

This layer comprises of several analytical techniques leading to solve the problem of semi wickedness of the release planning by meaningfully analyzing the wide variety of gathered and generated data. Some of these techniques were described in Section x.4.2. This platform is aligned with employed techniques and consists of modules which are adapted with three technology pillars employed in successful analytics platform [45]. Several silos of data are delivered as input to the data analytics platform. This data may have variety of sources such as former projects, expert's opinions, or similar projects. This layer is interpreting and structuring the data. The large-scale computing module is evaluating large sets of generated data as well as detecting inconsistencies in the data. The analysis component provides the results of analysis on solution space.

4.2 Analytical Techniques

With the different types of data retrieved from the various sources of information, the follow-up question is how to analyze them and create new insight. While there is a broad range of existing techniques available, there are a few of them that have been successfully used in the context of release planning. In what follows, we describe two of these techniques. In addition, we present the analytical dimension of the technique called *morphological analysis* and describe its usage in the context of (release planning) problem structuring.

4.2.1 Identification of Customer Segments from Applying Clustering [46]

A density-based clustering algorithm is used for identifying segments of customers having similar preference structures for product features. We consider a set of L customers represented by C = {c(1), c(2), …, c(L)}. A given set of M features represented by F = {feature(1), feature(2), …, feature(M)} is studied for their (complete) implementation in an evolving software system.

Definition. *Cluster configuration* is a partition of the set C into Q subsets. A cluster configuration is represented as CC(i) = {cc(i,1), …., cc(i,Q)} where cc(i,j) is the j-th cluster of customers in the i-th cluster configuration such that $\bigcup_{j=1\ldots Q} c\ (i,j) = C$ and cc(i,j_1) ∩ cc(i,j_2) = ∅ for all pairs of j_1 and j_2.

Multiple cluster configurations can be generated by changing input parameters to the clustering algorithm. In Figure 8, the cluster configuration CC(1) partitions the customers into two clusters cc(1,1) and cc(1,2).

Definition. For a given cluster configuration CC(i), a *product variant* p(i,j) is defined as the subset of the set of features F offered to the cluster of customers cc(i,j) where (1 ≤ j ≤ Q).

In this way, a company could offer one product variant per cluster of customers for all the cluster configurations.

Definition. *Product portfolio* is a set of product variants corresponding to a given cluster configuration. CC(i) corresponds to a product portfolio PP(i) = {p(i,1), …., p(i,Q)} such that $\bigcup_{j=1\ldots Q} p(i,j) = F$ and p(i,j_1) ∩ p(i,j_2) ≠ ∅ for all pairs of j_1 and j_2.

For example, the product portfolio PP(1) shown in Figure 3 contains two product variants corresponding to the two clusters in CC(1). The product variants differ in their feature sets but have feature(4) in common.

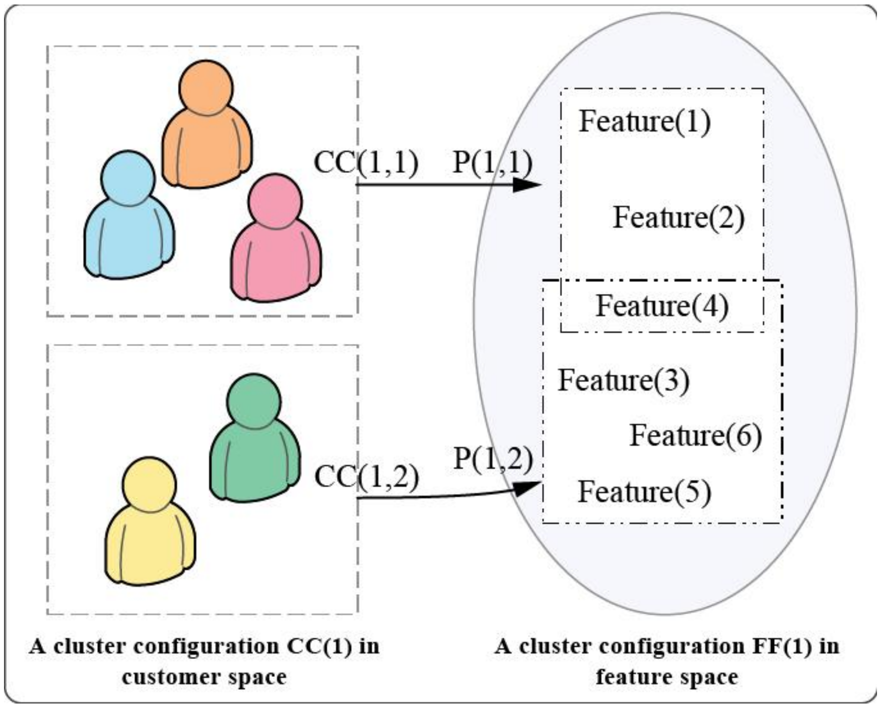

CC(1,1)
P(1,1)
Feature(1)
Feature(2)
Feature(4)
Feature(3)
Feature(6)
CC(1,2)
P(1,2)
Feature(5)
A cluster configuration CC(1) in customer space
A cluster configuration FF(1) in feature space

Figure 3. A cluster configuration and corresponding feature clusters

Following, we describe the assumptions for customer's clustering:

1. The number of clusters is not pre-defined as clustering is preformed based on preferences.
2. The customers represent organizations with different market shares and seeing different level of value in each product.

3. Having similar preferences for features of the product defines the cohesiveness of the clusters. Each member of a cluster should be within the maximum distance (predefined) of at least one other member of the same cluster.
4. Clusters are none overlapping, meaning that clusters should be at some distance from each other.

DBSCAN as a density-based spatial clustering algorithm for applications with noise [47] meets the first condition since it does not require the number of clusters as an input parameter. It also meets the third condition listed above through the neighborhood distance parameter. Fourth condition is also fulfilled since DBSCAN forms clusters with any arbitrary shape and they are separated by areas of low density (noise). However, DBSCAN treats all the customers as equally important. DBSCAN algorithm is used for identification of clusters of customers since it meets most of the conditions listed above.

Once we plan for product release, each segment of customers becomes a data point in the data set. Each data point is represented by a vector containing M values, where M1 is the number of features planned for the next release.

The major advantage of using DBSCAN is that clusters are formed only if there is sufficient level of cohesion amongst the data points in groups of customers representing unique market segments. Each market segment is mapped to a tentative product variant containing the features highly desired by customers.

4.2.2 Morphological Analysis

Morphological Analysis (MA) is a method for identifying, structuring and investigating the total set of possible relationships contained in a given multidimensional problem complex. MA allows small groups of subject specialists to define, link, and internally evaluate the parameters of complex problem spaces, creating a solution space and a flexible inference model [48] [49]. MA has been applied successfully in strategic planning and decision support in various domains such as governance of technological development and modelling the Bioethics of drug redevelopment which were reported more comprehensively by Ritchey [48].

MA provides an "if-then" laboratory within which drivers, and certain conditions can be assumed and range of associated solutions found, in order to test various inputs against possible outputs [50]. Generally, MA is intended to broaden the space of alternatives by systematic search for combination of attributes and systematically narrow them down through the results. The result of MA is called *a morphological field*. Morphological fields describe the total problem complex. MA consists of the following steps [50]:

Analysis phase

1. Extracting dimensions, parameters or variables, which define the essential nature of the problem complex or scenario.
2. Define a range of relevant, discrete values or conditions, for each variable.

Synthesize phase

3. Assess the internal consistency of all pairs of variable conditions
4. Synthesize an internally consistent outcome space.
5. Iterate the process if necessary.

In what follows, we provide the key concept and notation from MA which is needed to understand the rest of the paper.

Definition. A *morphological field* is a field of constructed dimensions or parameters which is the basis for a morphological model. This will shows by F(n,l) as

F(n,l) ∈ FEATURES | for all n , l where n ∈ {f| F is feature} refers to the functionality of the features, and l ∈ {L| L is level of functionality for M} indicates the level of functionality implemented.

Definition. *Cross Consistency Assessment* (CCA) pertains to the process by which the parameter values V(n,l) ∈ VALUE (or parameter conditions) in the morphological field are compared with each another. This process is intended reducing the total problem space to a smaller (internally consistent) solution spaces.

For diving deeper into the problem and examining internal relationships between field parameters [48], Cross Consistency Assessment (CCA) analysis is performed. This analysis acts as "garbage detector" and takes contradictory value pairs out of the solution space. Several types of relation between elements are detected. The data is extracted in two different stages firstly, utilizing stakeholder's expertise and secondly by using idea coming from the crowd. The result of CCA are the V(n,l) values inside the (symmetric) CCA matrix. The different values (illustrated and classified in Figure 4) are defined as follows:

- *Logical relation* in the forms of contradictions or dependency based on the nature of involved concepts,
- *Empirical constraints* in the forms of contradictions or dependency, which is not probable based on experience,
- *Normative constraints* in the forms of contradictions or dependency which is accepted as a contextual norm,

- *Incomparable relation* indicates for the type of relations that will not lead into any meaningful comparison as there is a difference in nature of the elements and,
- *Synergetic relation* indicates the cost or effort impact of implementing one feature on the other feature.

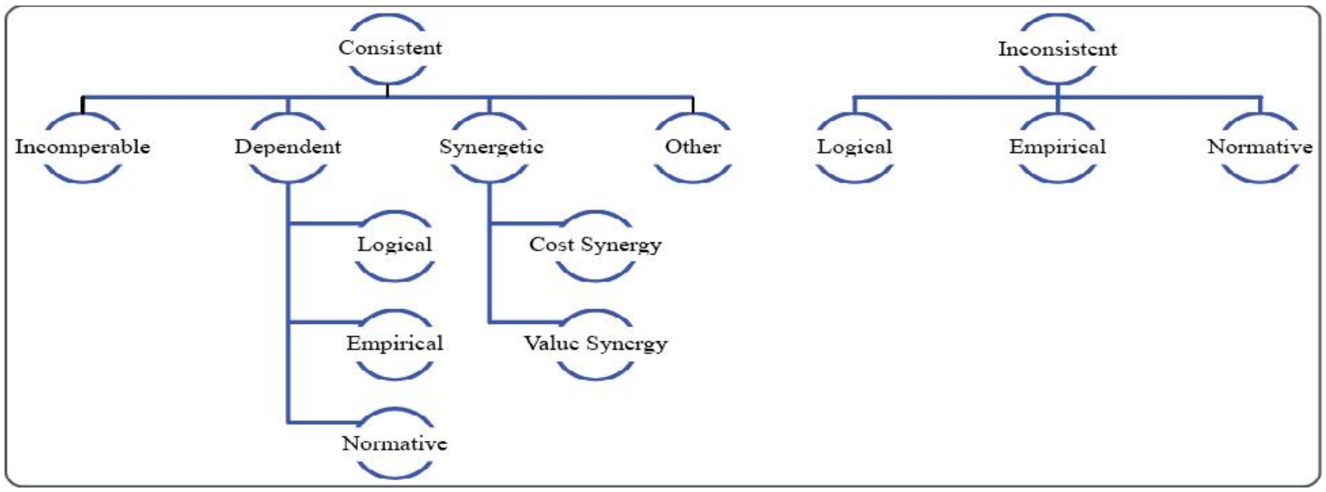


Figure 4. Hierarchy of relations in detected in Morphological Analysis

CCA analysis is providing a way to examine a set of parameters representing one context (i.e., decision criteria), against another set which represents another context (i.e., features) and is providing a tool for presenting one modelling context set as input and another set as output. There are different and conflicting criteria influencing release decisions [51]. At this step, by defining a solution set as an input, the probable output solutions on the other side are studied.

In the following, we summarize the usage of MA related to these sub-problems.

Definition of planning criteria: Providing and maintaining close relation with the market in order to first understand the needs and second, audit the response of the market to the offers by utilizing open innovation approach followed by MA (sub-problems 2 to 4).

Definition of utility function: Defining utility function based on important criteria of decision making in the context of project criteria and then assessing the utility of having set of the extracted criteria (sub-problem 1) supported by open innovation platform followed by MA.

Elicitation and structuring of dependencies: Detection and formulation of dependencies are supported by performing CCA. Pairwise consistency assessment between features shows the explicit dependencies and synergies, as well as hidden dependencies and synergies between features.

Feature prioritization: Prioritization of features towards given set of criteria.

Objectives of plan evaluation: Defining resource and quality as objectives of plan evaluation, helps in evaluating plans toward these considerations.

Table 4. Applications of MA for release planning

#	Sub-problem	MA Application	Extracted Data
1	Planning Criteria	Criteria modeling & assessment	Decision making criteria for selecting features
2	Features elicitation	Feature model & assessment	Influential features on choosing a product
3	Feature dependency and synergy elicitation	Cross Consistency Assessment	Dependencies and synergies between features
4	Prioritization of features	Feature-criteria relational model & assessment	Prioritization of features towards a set of consistent criteria
5	Plan evaluation	Object modeling & assessment	Objectives of plan evaluation

5. Analytical Release Planning – A Case Study

This section demonstrates the usage and customization of the AOI approach. While applicable for all the classes of release planning problems discussed in Section x.2, in what follows, we describe the application of AOI for two of them in more detail. Before that, we present the context and the content of the case study.

5.1 OTT Case Study – The Context and Content

The case study is in the context of deciding for a real over the top TV (OTT) features offered by a service provider. Over the top TV is referred to the media provided over the internet apart from operator infrastructure to distribute the content and putting the responsibility of media transportation on ISP side [52, 53].

Company X has a new OTT product with various features which were initially extracted from market expectations and by doing research on similar products in other markets. The stakeholders estimate the cost of implementing each feature. The company aims to extract sets of services for each quarter of the year, ensuring the best income and largest set of customer. To do so, they gather customer's willingness to pay for each feature from current customers (via customer management system (CMS) surveys) and potential customers (via crowdsourcing). The utility for the company is to achieve maximum income while maintaining their market share. In order to perform crowdsourcing, the HIT submitted to Amazon Mechanical Turk was targeting 100 workers. Also some context specific validation is done on this 100 individuals including metrics to indicate the truth worthiness of their answers (these metrics are defined by AMT and include ones such as expertise of worker, worker's score, number of rejected tasks …) 10 out of these 100 were not validated due to these issues (this is shown as a step in Figure 5).

5.2 Formalization of the Problem

The problem studied as part of the case study is a variant of the What-to-release problem including advance feature dependencies as outlined in Sub-section x.2.1. In what follows, we give a more detailed formal description of the problem. For that, we consider a set of features called FEATURE. F(n,l) $\in$ FEATURE presents a specific feature in which n is the feature number and l is the functionality level. These features are planned over *K* release where *k = 1…n*, and each release has a release weight ∈ *{1,…,9}*.

A release plan is the assignment of features F(n,l) $\in$ FEATURES at functionality level l to one of the upcoming releases (or deciding to postpone the feature):

$$x(n,l)=:\begin{cases} \text{k} & \text{if F(n, l) is offered at relase k} \\ \quad 0 & \text{otherwise} \end{cases} \tag{1}$$

Each feature to be provided causes certain amount of (fixed) cost abbreviated by F_cost(n,l).

From the stakeholders' perspective each F(n,l) ∈ FEATURE has a cost of implementation, the weighted average of stakeholders' prediction (and considering the person's importance) is used in the model:

$$F\_C \quad (n,l) = \frac{\sum_{s=1\ldots S} Es \quad (n,l,s) \times in \quad (s)}{\sum_{s=1\ldots S} in \quad (s)} \tag{2}$$

Assuming capacities budget(k) for the different time periods k (e.g., quarters of a year), the budget constraint related to release k is formulated as:

$$\sum_{n,l:\, x(n,l)=k} F\_c \quad (n,l) \times x(n,l) \le b \quad (k) \tag{3}$$

Also, feature dependency constraints need to be considered for release decision making. The detailed constraints are presented in Appendix 1. Next, we define the objectives of planning. For each F(n,l) ∈ FEATURE we assume that we have information about the willingness to pay (defined as a monthly fee) from two groups of customers: (i) current customers of the company and (ii) potential new customers. The median of the data collected is used in order to make the data less affected by skewed and outlier data.

$$F\_W \quad (n,l) = m \quad (W\ ll \quad (n,l)) \tag{4}$$

Problem. For a set of candidate features FEATURE (which includes different types of features and their related functionality level), the problem is to:

(i) Find a subset of features named feat* ⊆ FEATURES which is of maximum overall utility and

(ii) Assign all F(n,l) ∈ feat* to a release where the feature will be offered.

The utility function is as follows:

$$Utility = \sum_{n,l:x(n,l)\neq 0} W \quad ht(x(n,l)) \times F\_W \qquad (n,l) \times w \quad ht\_C \qquad (x(n,l)) \\ Utility \rightarrow Max! \tag{5}$$

5.3 The Case Study Process

The detailed case study process is described in Figure 5. The process consists of 16 steps, utilizing the different parts of the AOI platform are described in Section x.4.3.

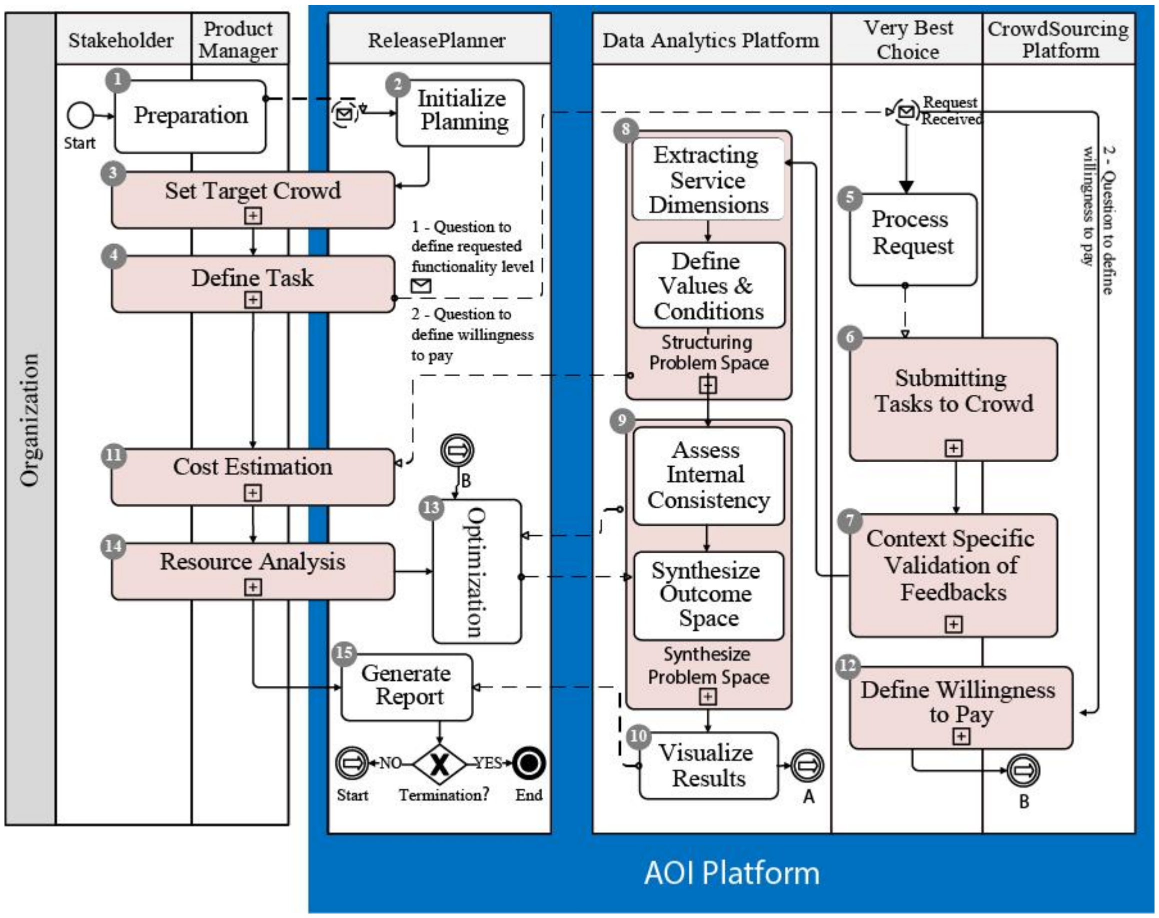


Figure 5. Process steps performed in the case study

Steps 1, 2, and 3 are initializing the project by defining the decision making criteria, inviting stakeholders and defining the special crowd needs, respectively. In our case, the crowd was constituted from individuals living in North America. In order to extract features and desirable levels of functionality (Step 4), Task 1 and Task 2 were submitted to the crowd:

Task 1: What are the features you are looking for in an OTT service?

Task 2: Which level of quality (degree of functionalities) do you expect from OTT services?

Amazon Mechanical Turk [42] is used as a micro-task market to establish the collaboration with the crowd. In order to manage collaboration between systems Very Best Choice [43] is used as the lightweight decision support system. This system is designed to facilitate proper priority decision making based on comprehensive stakeholder involvement. In this case, two groups of stakeholders, namely technical and managerial experts, are involved. Tasks 1 and 2 were assigned to 90 participants. The extracted features and their rankings are shown in Figure 6.

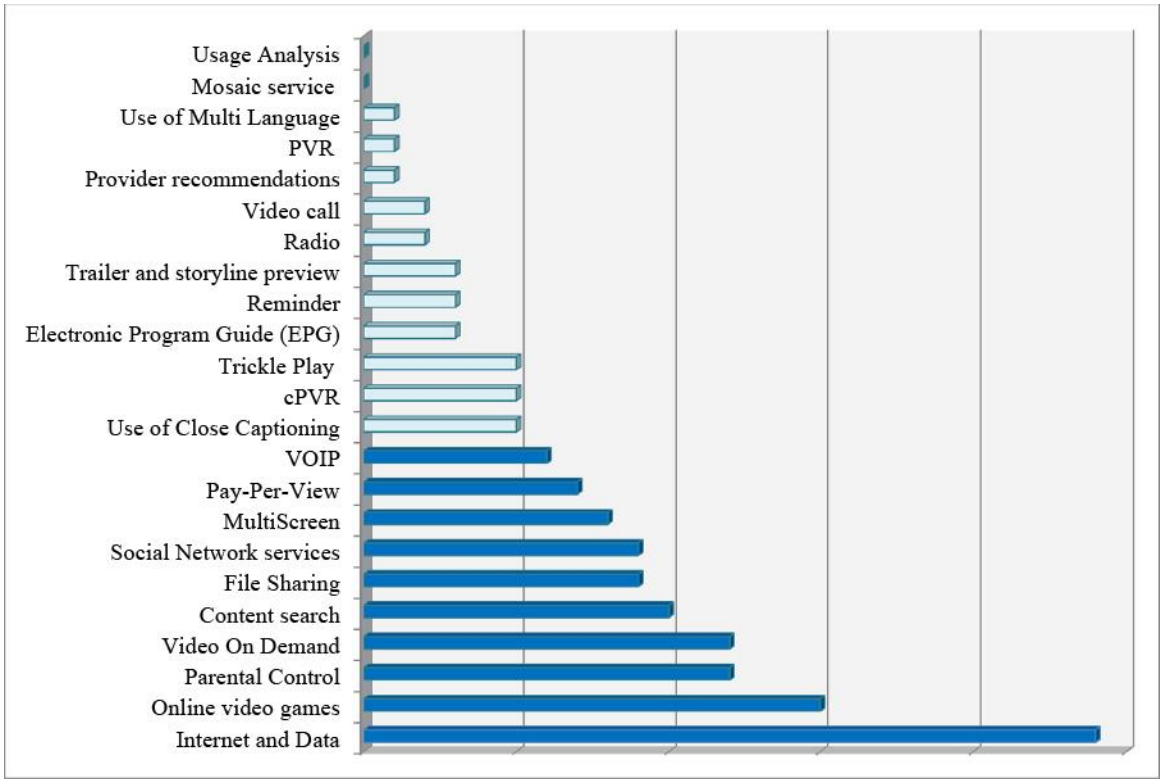


Figure 6. Features extracted from crowdsourcing and their ranking. The top ten most attractive (dark) ones were selected for the case study

Context specific validation and applying qualification type, such as "degree of experience based on acceptance rate of the worker at the Amazon Mechanical Turk" is applied in Step 7. The top ten claimed features are used to describe the case study. The features and their related levels of functionality are defined as below:

F(Online video game,l_1)| l_1 ∈{Paid, Free}
F(VOIP,l_2)| l_2 ∈{Yes, No}
F(Social network,l_3)| l_3 ∈{Rate, Comment, FB integration, twitter integration}
F(Video on demand,l_4)| l_4 ∈{Yes, No}
F(Parental control,l_5)| l_5 ∈{Basic, Advanced, Premium}
F(Content search,l_6)| l_6 ∈{Basic, Advanced, Premium}
F(File sharing,l_7)| l_7 ∈{Limited, Limited Chargeable, Unlimited}
F(Pay-per-view,l_8)| l_8 ∈{Yes, No}
F(Internet and data,l_9)| l_9 ∈{Limited, Limited Chargeable, Unlimited}
F(Multiscreen,l_{10})| l_{10} ∈{Basic, Advanced, Premium}

For ease of notation, we have used F_1, F_2 ... F_{10} (Sequentially in the above list) for expressing all selected features including their different levels of functionality. Step 8 is designed to extract features and their related functionality levels. Extracting features and defining the content of a morphological box is an attempt to further structure the release planning problem space.

5.4 Release planning in the presence of advanced feature dependencies and synergies

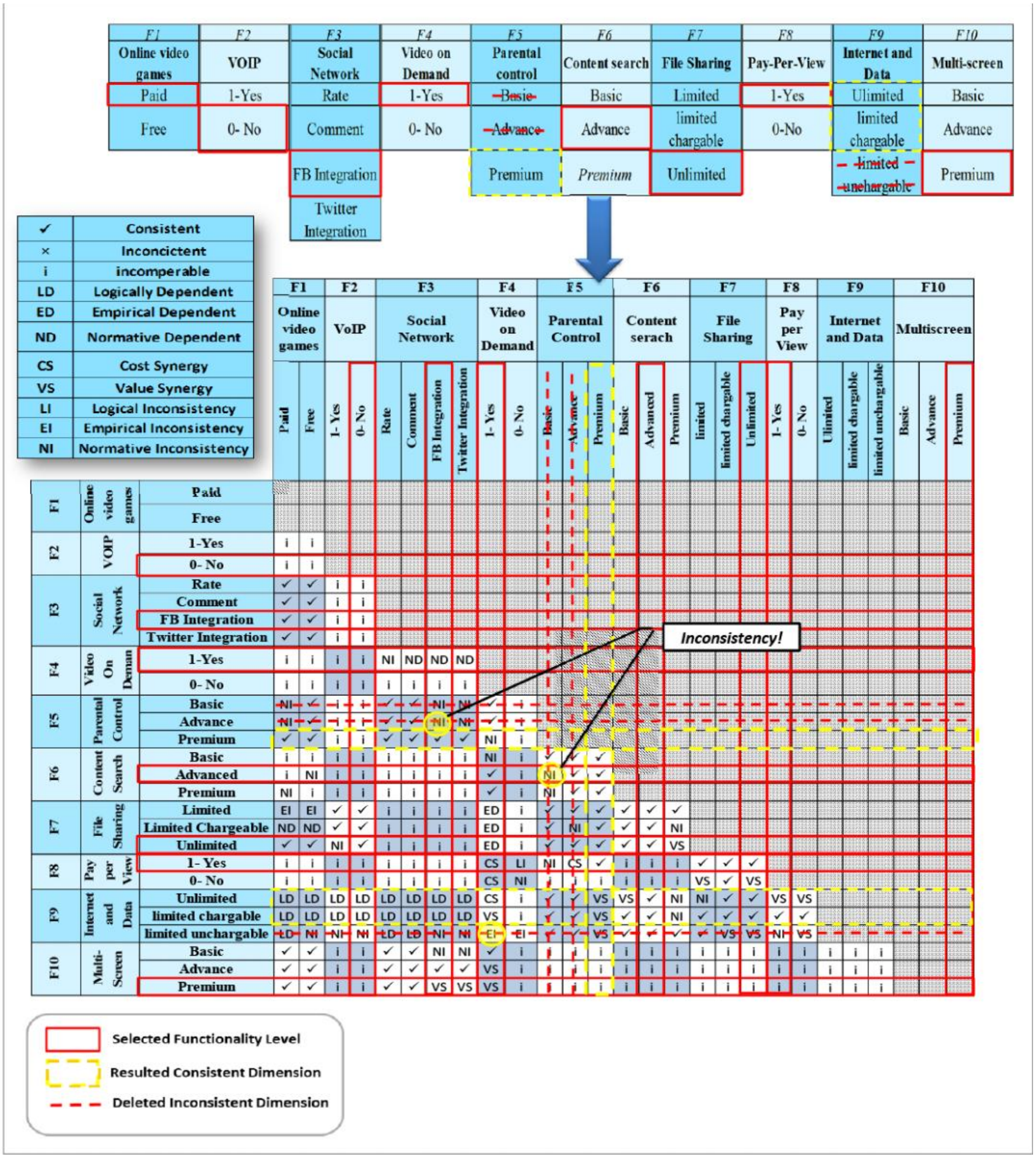
Step 9 in this case study process, is designed to detect service dependencies and inconsistencies as well as extracting cost and value synergies. All these relation between features are taken as an input for the subsequent release optimization process. In addition to inconsistency analysis, cost and value synergy relationships between features' functionality levels offered in conjunction are taken as input for optimizing release plans. Some examples are presented below:

Example: x(parental control, Basic) NAND x(Online video games, Paid) means that x(parental control, Basic) = 0 ‖ x(Online video games, Paid) = 0

Example: F(Multi-Screen, Premium) becomes 30% less expensive if offered no earlier than F(Multi-Screen, Basic).

Example: The sum value of {F(Online Video Gaming, Premium), F(Parental Control, Premium)} is increased by 25% if these items are all offered in the same release.

The values extracted during the process of assessment are applied in Figure 7 and further analyzed to extract the inconsistencies. Without performing CCA, the proposed feature implementations would violate these constraints and thus creating customer and user concerns.



F1	*F2*	*F3*	*F4*	*F5*	*F6*	*F7*	*F8*	*F9*	*F10*
Online video games	VOIP	Social Network	Video on Demand	Parental control	Content search	File Sharing	Pay-Per-View	Internet and Data	Multi-screen
Paid	1-Yes	Rate	1-Yes	~~Basic~~	Basic	Limited	1-Yes	Ulimited	Basic
Free	0- No	Comment	0- No	~~Advance~~	Advance	limited chargable	0-No	limited chargable	Advance
		FB Integration		Premium	*Premium*	Unlimited		~~limited unchargable~~	Premium
		Twitter Integration							

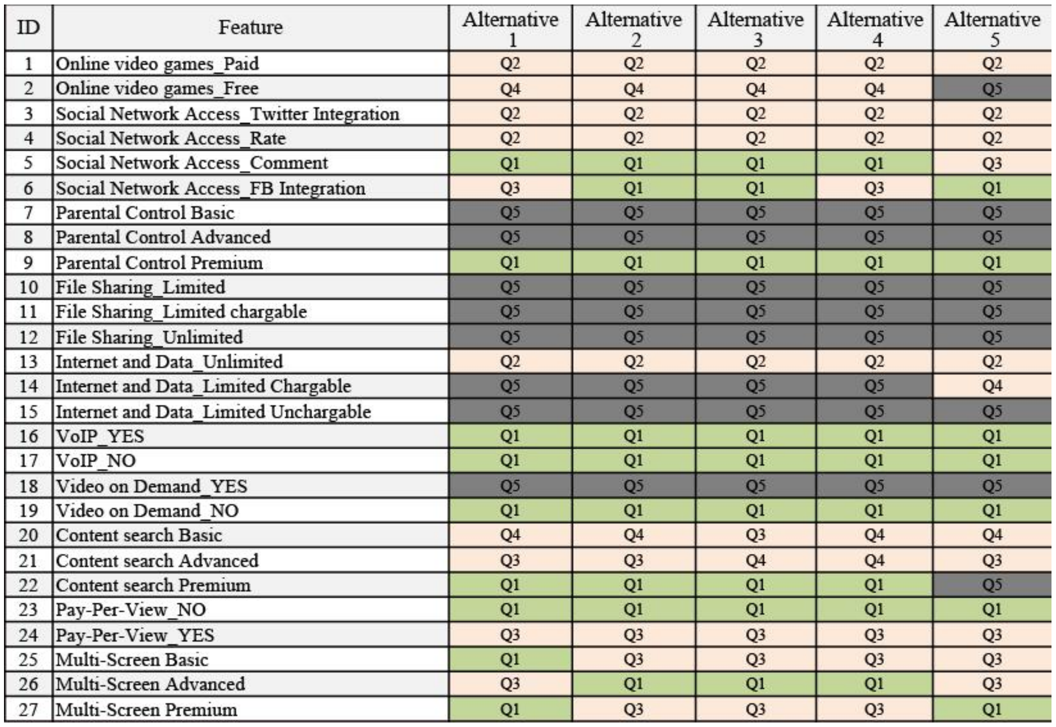
✓	Consistent
×	Inconcictent
i	incomperable
LD	Logically Dependent
ED	Empirical Dependent
ND	Normative Dependent
CS	Cost Synergy
VS	Value Synergy
LI	Logical Inconsistency
EI	Empirical Inconsistency
NI	Normative Inconsistency

		F1		F2		F3				F4		F5			F6			F7			F8		F9			F10		
		Online video games		VoIP		Social Network				Video on Demand		Parental Control			Content serach			File Sharing			Pay per View		Internet and Data			Multiscreen		
		Paid	Free	1- Yes	0- No	Rate	Comment	FB Integration	Twitter Integration	1- Yes	0- No	~~Basic~~	~~Advance~~	Premium	Basic	Advanced	Premium	limited	limited chargable	Unlimited	1- Yes	0- No	Ulimited	limited chargable	limited unchargable	Basic	Advance	Premium
F1 Online video games	Paid																											
	Free																											
F2 VOIP	1-Yes	i	i																									
	0- No	i	i																									
F3 Social Network	Rate	✓	✓	i	i																							
	Comment	✓	✓	i	i																							
	FB Integration	✓	✓	i	i																							
	Twitter Integration	✓	✓	i	i																							
F4 Video On Deman	1-Yes	i	i	i	i	NI	ND	ND	ND																			
	0- No	i	i	i	i	i	i	i	i																			
F5 Parental Control	Basic	~~NI~~	~~✓~~	~~i~~	~~i~~	~~✓~~	~~✓~~	~~NI~~	~~NI~~	~~✓~~	~~i~~																	
	Advance	~~NI~~	~~✓~~	~~i~~	~~i~~	~~✓~~	~~✓~~	~~NI~~	~~NI~~	~~✓~~	~~i~~																	
	Premium	✓	✓	i	i	✓	✓	✓	✓	NI	i																	
F6 Content Search	Basic	i	i	i	i	i	i	i	i	NI	i	✓	✓	✓														
	Advanced	i	NI	i	i	i	i	i	i	✓	i	NI	✓	✓														
	Premium	NI	i	i	i	i	i	i	i	✓	i	NI	✓	✓														
F7 File Sharing	Limited	EI	EI	✓	✓	i	i	i	i	ED	i	✓	✓	✓	✓	✓	✓											
	Limited Chargeable	ND	ND	✓	✓	i	i	i	i	ED	i	✓	NI	✓	✓	✓	NI											
	Unlimited	✓	✓	NI	✓	i	i	i	i	ED	i	✓	✓	✓	✓	✓	VS											
F8 Pay per View	1- Yes	i	i	i	i	i	i	i	i	CS	LI	NI	CS	✓	i	i	i	✓	✓	✓								
	0- No	i	i	i	i	i	i	i	i	CS	NI	i	i	i	i	i	i	VS	✓	VS								
F9 Internet and Data	Unlimited	LD	LD	LD	LD	LD	LD	LD	LD	CS	i	✓	✓	VS	VS	✓	NI	NI	✓	✓	VS	VS						
	limited chargable	LD	LD	LD	LD	LD	LD	LD	LD	VS	i	✓	✓	VS	✓	✓	NI	✓	✓	✓	✓	✓						
	limited unchargable	~~LD~~	~~NI~~	~~NI~~	~~NI~~	~~LD~~	~~LD~~	~~NI~~	~~NI~~	~~EI~~	~~EI~~	~~✓~~	~~✓~~	~~VS~~	~~✓~~	~~✓~~	~~✓~~	~~✓~~	~~VS~~	~~VS~~	~~NI~~	~~VS~~						
F10 Multi-Screen	Basic	✓	✓	i	i	✓	✓	NI	NI	✓	i	i	i	i	i	i	i	i	i	i	i	i	i	i	i			
	Advance	✓	✓	i	i	✓	✓	✓	✓	VS	i	i	i	i	i	i	i	i	i	i	i	i	i	i	i			
	Premium	✓	✓	i	i	✓	✓	VS	VS	VS	i	i	i	i	i	i	i	i	i	i	i	i	i	i	i			

🟥	Selected Functionality Level
▭	Resulted Consistent Dimension
- - -	Deleted Inconsistent Dimension

Figure 7. CCA analysis of top 10 features and their functionalities via Crowdsourcing

The estimation process is initiated for an agreed upon criterion (willingness to pay). Each feature and the different levels of functionality are evaluated on a nine-point scale ranging from “extremely low” to “extremely high” (corresponding to Step 12 in Figure 5.)



ID	Feature	Alternative 1	Alternative 2	Alternative 3	Alternative 4	Alternative 5
1	Online video games_Paid	Q2	Q2	Q2	Q2	Q2
2	Online video games_Free	Q4	Q4	Q4	Q4	Q5
3	Social Network Access_Twitter Integration	Q2	Q2	Q2	Q2	Q2
4	Social Network Access_Rate	Q2	Q2	Q2	Q2	Q2
5	Social Network Access_Comment	Q1	Q1	Q1	Q1	Q3
6	Social Network Access_FB Integration	Q3	Q1	Q1	Q3	Q1
7	Parental Control Basic	Q5	Q5	Q5	Q5	Q5
8	Parental Control Advanced	Q5	Q5	Q5	Q5	Q5
9	Parental Control Premium	Q1	Q1	Q1	Q1	Q1
10	File Sharing_Limited	Q5	Q5	Q5	Q5	Q5
11	File Sharing_Limited chargable	Q5	Q5	Q5	Q5	Q5
12	File Sharing_Unlimited	Q5	Q5	Q5	Q5	Q5
13	Internet and Data_Unlimited	Q2	Q2	Q2	Q2	Q2
14	Internet and Data_Limited Chargable	Q5	Q5	Q5	Q5	Q4
15	Internet and Data_Limited Unchargable	Q5	Q5	Q5	Q5	Q5
16	VoIP_YES	Q1	Q1	Q1	Q1	Q1
17	VoIP_NO	Q1	Q1	Q1	Q1	Q1
18	Video on Demand_YES	Q5	Q5	Q5	Q5	Q5
19	Video on Demand_NO	Q1	Q1	Q1	Q1	Q1
20	Content search Basic	Q4	Q4	Q3	Q4	Q4
21	Content search Advanced	Q3	Q3	Q4	Q4	Q3
22	Content search Premium	Q1	Q1	Q1	Q1	Q5
23	Pay-Per-View_NO	Q1	Q1	Q1	Q1	Q1
24	Pay-Per-View_YES	Q3	Q3	Q3	Q3	Q3
25	Multi-Screen Basic	Q1	Q3	Q3	Q3	Q3
26	Multi-Screen Advanced	Q3	Q1	Q1	Q1	Q3
27	Multi-Screen Premium	Q1	Q3	Q3	Q3	Q1

Figure 8. Optimized and diversified release plans in fulfilment of all types of dependencies and utilizing synergies

In addition, customers are requested to articulate their willingness to pay for each functionality level of features. In Step 13 of the process, data from the two separate customer groups are gathered, and potential customers' ideas are investigated via crowdsourcing while the data from current users are extracted from customer service surveys of the company. A task was submitted to crowd in order to indicate their willingness to pay. For each functionality level, 50 participants completed the task via crowdsourcing, indicating the potential customer's willingness. Also the willingness to pay is investigated among the current customers via customer service surveys. Having all the data needed, in Step 15, ReleasePlanner is generating the alternative plans shown in Figure 8. Each column represents a plan alternative, and each row corresponds to a feature. The table entries of Figure 8 describe that the respective feature is offered in Q1, Q2, Q3, and Q4 or that it is postponed (Q5).

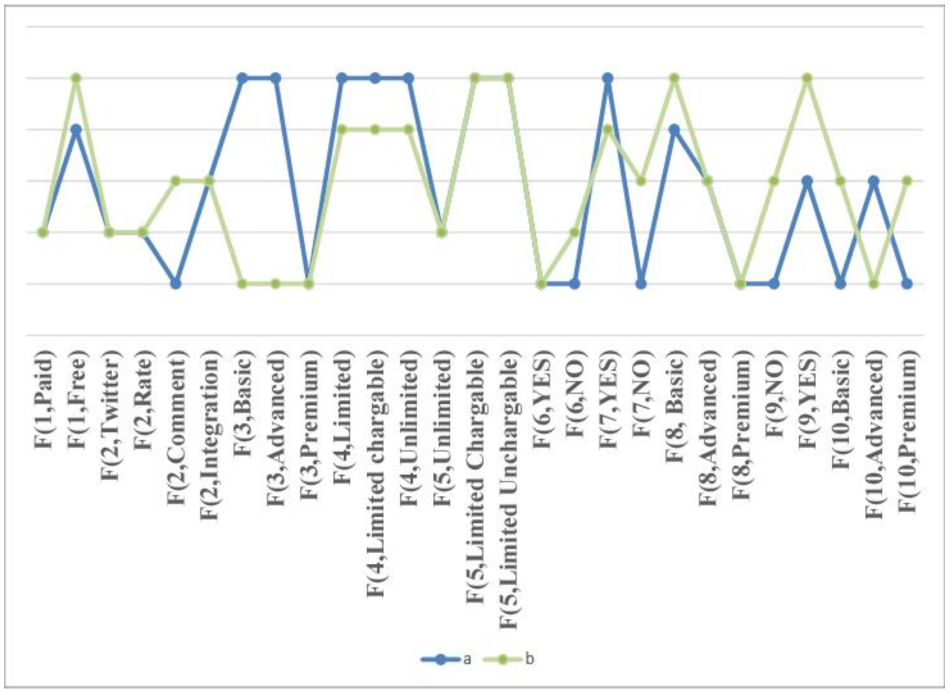
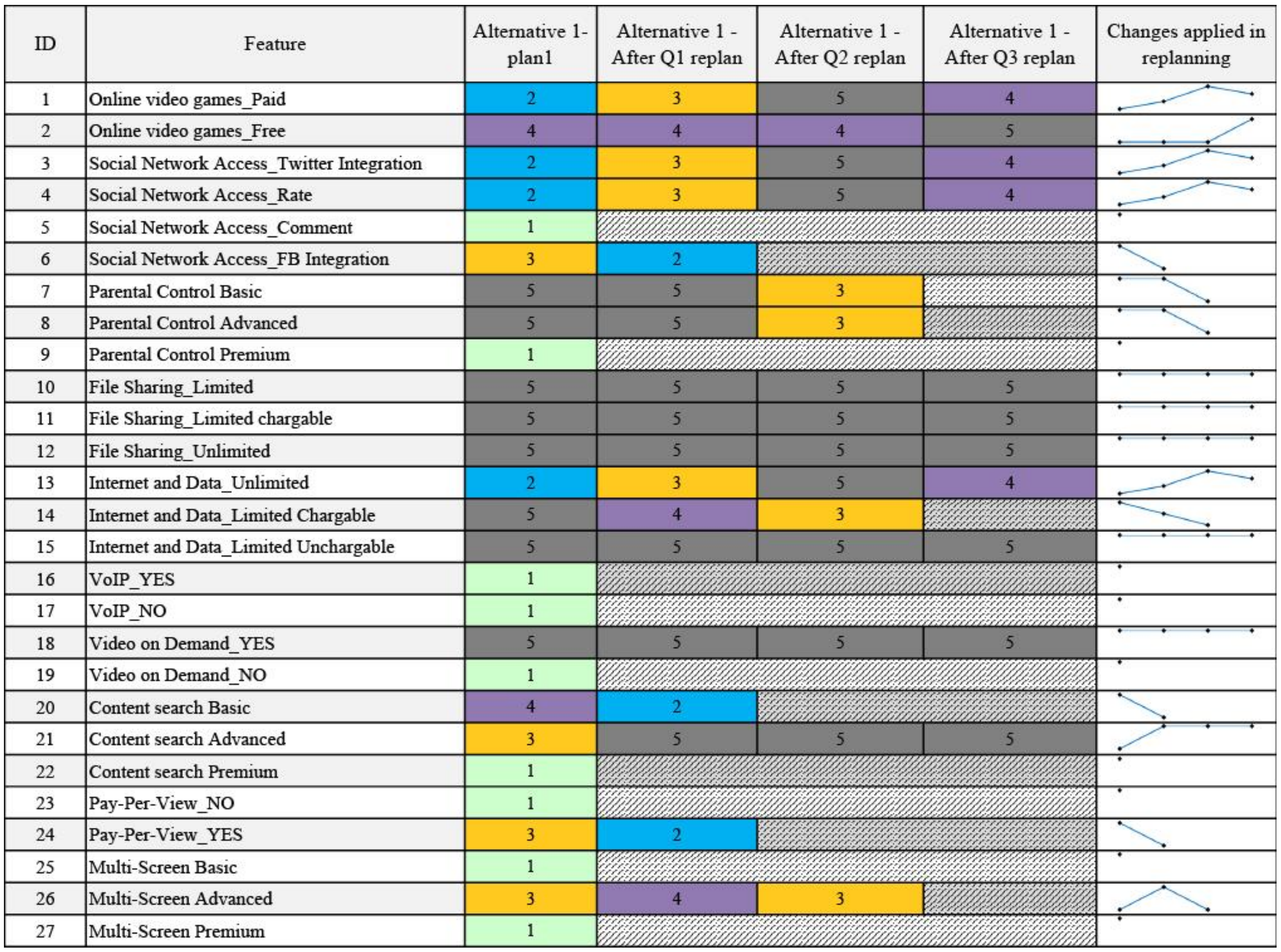


Figure 9. Comparison of the structure between optimized release plans (a) with and (b) without synergies. The placements of dotes on horizontal lines, are showing the release quarter for each feature (Q1 to Q5)

Ignoring cost and value synergies ignores potential resource savings and additional value creation opportunities. Figure 9 shows the structural changes of release plans while we consider synergies between features. As it is shown in Figure 9, considering synergies will affect the planning results. Besides, they not only change the offered feature but they also affect the release value presented as the stakeholder's satisfaction; the value improvement is more than 10% in the discussed case.

5.5 Real-time what-to-release planning

Incrementally building and deploying products, relying on deep customer insight and real-time feedback is expected to create products with a higher customer hit rate and faster development. The process for achieving these goals needs to be based upon continuously up-to-date and most relevant information and deep and comprehensive data analytics to utilize business and market patterns, trends and predictions. Short feedback cycles need to be performed, with the goal of evaluating the attractiveness of features and their combination. After each quarter of the year, with one set of features implemented, we created synthetic data for both customer groups to simulate real-time changes of customer value and priorities.



ID	Feature	Alternative 1-plan1	Alternative 1 - After Q1 replan	Alternative 1 - After Q2 replan	Alternative 1 - After Q3 replan	Changes applied in replanning
1	Online video games_Paid	2	3	5	4	
2	Online video games_Free	4	4	4	5	
3	Social Network Access_Twitter Integration	2	3	5	4	
4	Social Network Access_Rate	2	3	5	4	
5	Social Network Access_Comment	1	▨	▨	▨	
6	Social Network Access_FB Integration	3	2	▨	▨	
7	Parental Control Basic	5	5	3	▨	
8	Parental Control Advanced	5	5	3	▨	
9	Parental Control Premium	1	▨	▨	▨	
10	File Sharing_Limited	5	5	5	5	
11	File Sharing_Limited chargable	5	5	5	5	
12	File Sharing_Unlimited	5	5	5	5	
13	Internet and Data_Unlimited	2	3	5	4	
14	Internet and Data_Limited Chargable	5	4	3	▨	
15	Internet and Data_Limited Unchargable	5	5	5	5	
16	VoIP_YES	1	▨	▨	▨	
17	VoIP_NO	1	▨	▨	▨	
18	Video on Demand_YES	5	5	5	5	
19	Video on Demand_NO	1	▨	▨	▨	
20	Content search Basic	4	2	▨	▨	
21	Content search Advanced	3	5	5	5	
22	Content search Premium	1	▨	▨	▨	
23	Pay-Per-View_NO	1	▨	▨	▨	
24	Pay-Per-View_YES	3	2	▨	▨	
25	Multi-Screen Basic	1	▨	▨	▨	
26	Multi-Screen Advanced	3	4	3	▨	
27	Multi-Screen Premium	1	▨	▨	▨	

Figure 10. Initial plan (first column) and evolution of plans over the time considering re-evaluation of feature values. The structural changes of features within plans are summarized in the most right column

The individual and collective value of features is hard to predict. The value depends on a number of factors which themselves are dynamically changing (e.g., competition, market trends, user acceptance). As trend of planning is shown in Figure 10, the features' values are changed during time based on stakeholders' point of view. In Figure 10, the structural changes of features in release plans are shown in the most right column. One circle shows that, the feature was released in the first cycle and one line demonstrates that the release plan has not changed over time for a specific feature

Figure 11 summarizes the changes caused by updated feature cost estimates and value predictions for the case that re-planning is conducted at the beginning of each quarter. Accommodating the most recent project information increases the validity and value creation capability of the generated release plans.

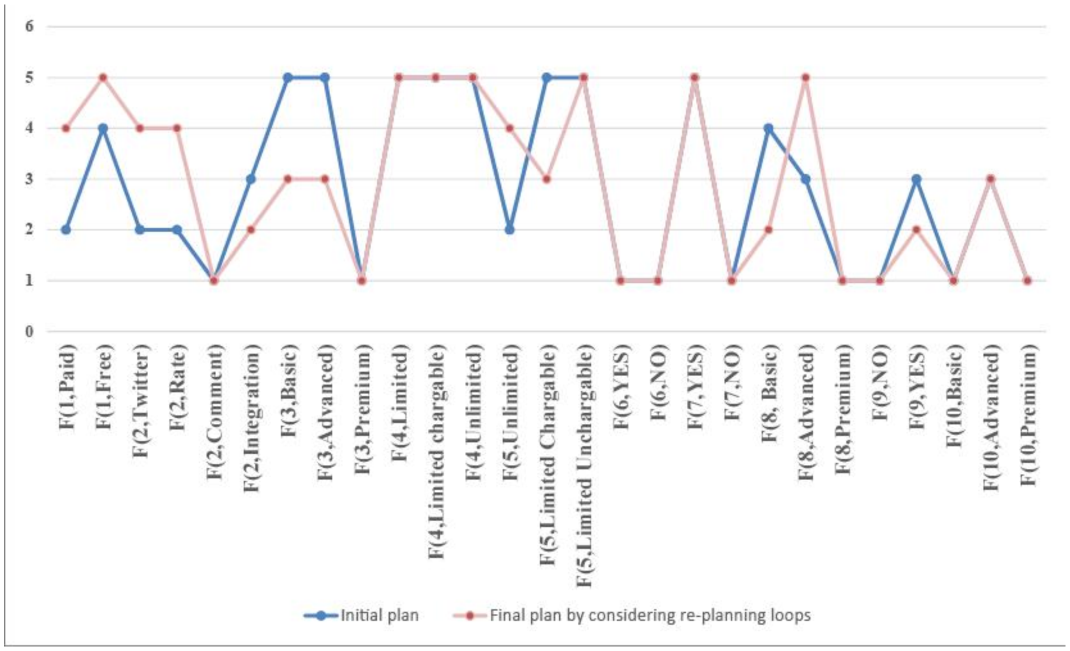


Figure 11. Comparison of the initial plan with implemented plan after considering three re-planning loops. The placements of dotes on horizontal lines compare the release quarter for each feature being issued (initial plan versus plan after three revisions)

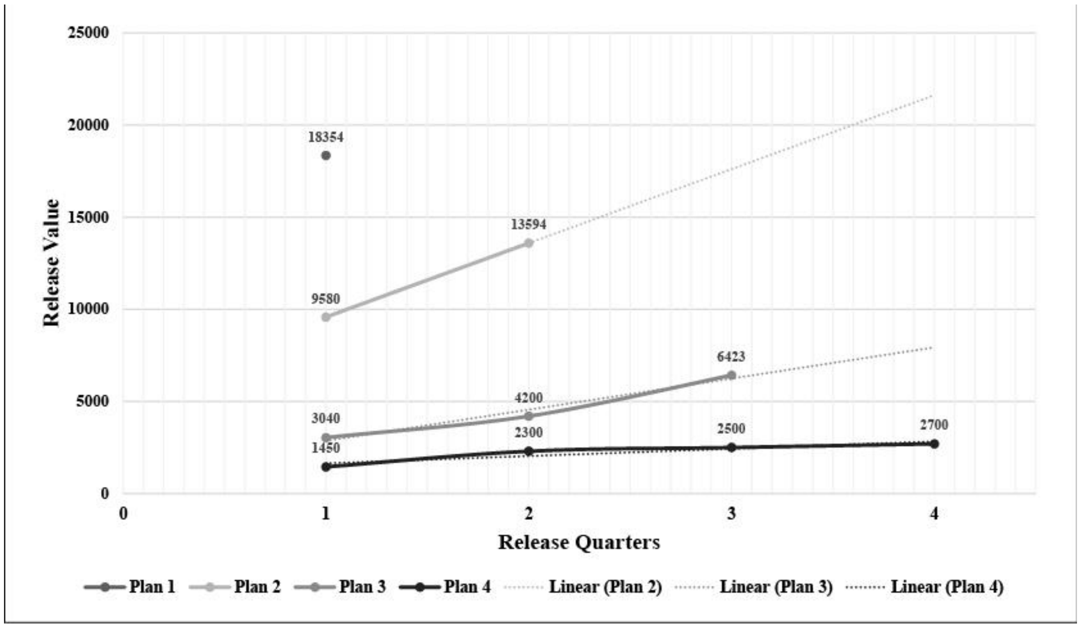
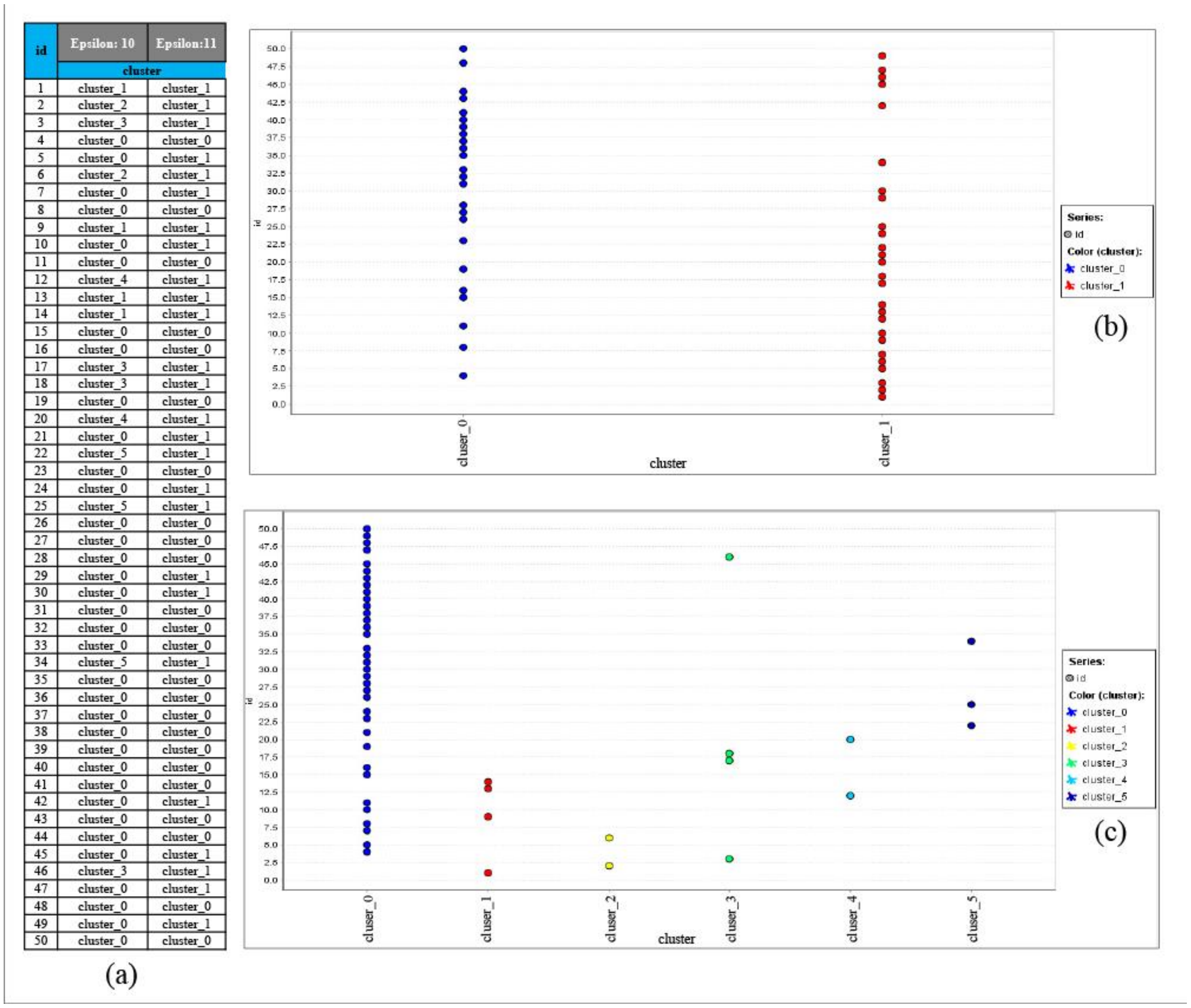


Figure 12. The increase of release value based on total number of stakeholder feature points by re-planning. The linear forecast of the value is shown for each level.

Re-planning adds significant value to the release plans and completely changes the structure of plans. The change of value during the four quarters of our planning is shown in Figure 12. In this figure, Plan 1 shows the values of all four releases at the initial point of planning. Plan 2 is calculated after implementing the first release plan and when the priorities and capacities were updated and re-planning conducted. The same process was applied for Plan 3 and Plan 4 at the end of implementing Q2 and Q3. This is mainly derived by the need to refine market needs to address uncertainty in the customers' needs and detecting the market trends.

5.6 Re-planning based on crowd clustering

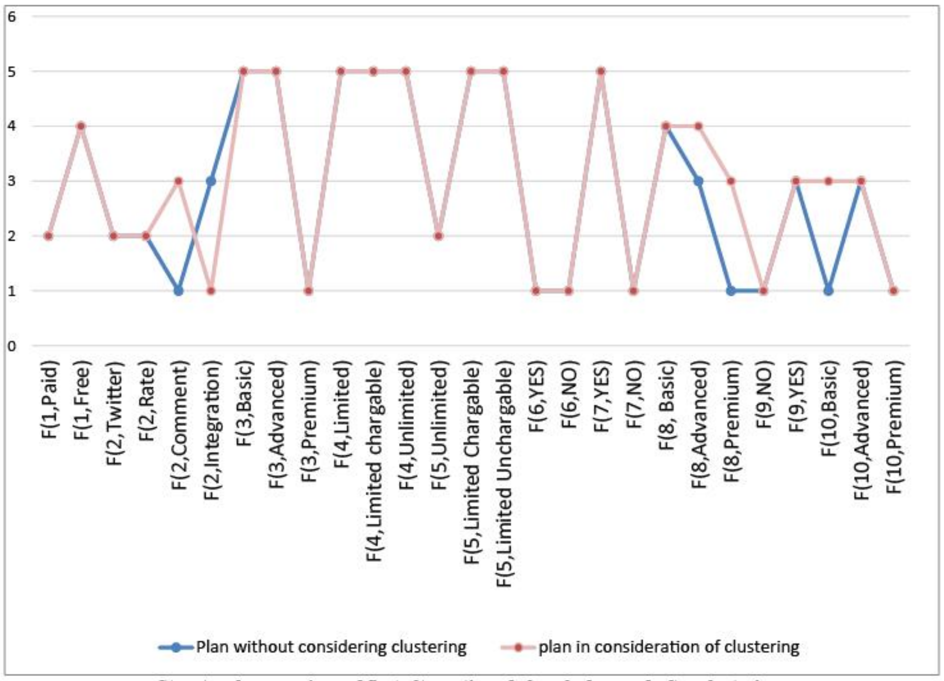
The cases studied in Sections 5.3, Section 5.4 and Section 5.5 were presented by gathering the preferences of two different customer groups and utilizing the median of the preferences to eliminate the skewed and outlier data. The intention of AOI is to gather and maintain the various types of relations and satisfy different groups of potential and current customers.



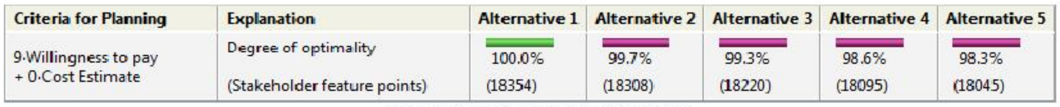
id	Epsilon: 10	Epsilon:11
	cluster	
1	cluster_1	cluster_1
2	cluster_2	cluster_1
3	cluster_3	cluster_1
4	cluster_0	cluster_0
5	cluster_0	cluster_1
6	cluster_2	cluster_1
7	cluster_0	cluster_1
8	cluster_0	cluster_0
9	cluster_1	cluster_1
10	cluster_0	cluster_1
11	cluster_0	cluster_0
12	cluster_4	cluster_1
13	cluster_1	cluster_1
14	cluster_1	cluster_1
15	cluster_0	cluster_0
16	cluster_0	cluster_0
17	cluster_3	cluster_1
18	cluster_3	cluster_1
19	cluster_0	cluster_0
20	cluster_4	cluster_1
21	cluster_0	cluster_1
22	cluster_5	cluster_1
23	cluster_0	cluster_0
24	cluster_0	cluster_1
25	cluster_5	cluster_1
26	cluster_0	cluster_0
27	cluster_0	cluster_0
28	cluster_0	cluster_0
29	cluster_0	cluster_1
30	cluster_0	cluster_1
31	cluster_0	cluster_0
32	cluster_0	cluster_0
33	cluster_0	cluster_0
34	cluster_5	cluster_1
35	cluster_0	cluster_0
36	cluster_0	cluster_0
37	cluster_0	cluster_0
38	cluster_0	cluster_0
39	cluster_0	cluster_0
40	cluster_0	cluster_0
41	cluster_0	cluster_0
42	cluster_0	cluster_1
43	cluster_0	cluster_0
44	cluster_0	cluster_0
45	cluster_0	cluster_1
46	cluster_3	cluster_1
47	cluster_0	cluster_1
48	cluster_0	cluster_0
49	cluster_0	cluster_1
50	cluster_0	cluster_0

Figure 13. (a) Result of clustering of 50 individuals with three different epsilons (b) Clustering results with ε=10 (c) Clustering results with ε = 11

Based on the approach presented in Section x.4.2.1, we cluster individuals inside the crowd with different neighborhood distance (Epsilon). By choosing epsilon as 10, the individuals inside the crowd are clustered into 6 categories. By choosing neighborhood distance as 11, clustering resulted in 2 categories. In order to investigate the effect of clustering on product planning, the impact of re-planning is shown by comparing the results of planning based on 2 versus 6 clusters (the clusters are shown in Figure 13).

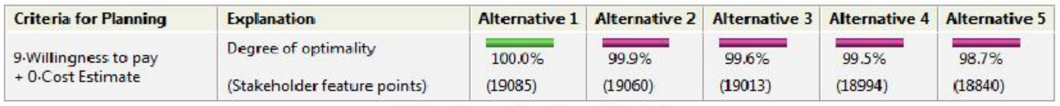


Structural comparison of first alternative of plans before and after clustering



Criteria for Planning	Explanation	Alternative 1	Alternative 2	Alternative 3	Alternative 4	Alternative 5
9·Willingness to pay + 0·Cost Estimate	Degree of optimality (Stakeholder feature points)	100.0% (18354)	99.7% (18308)	99.3% (18220)	98.6% (18095)	98.3% (18045)

Value without considering clustering



Criteria for Planning	Explanation	Alternative 1	Alternative 2	Alternative 3	Alternative 4	Alternative 5
9·Willingness to pay + 0·Cost Estimate	Degree of optimality (Stakeholder feature points)	100.0% (19085)	99.9% (19060)	99.6% (19013)	99.5% (18994)	98.7% (18840)

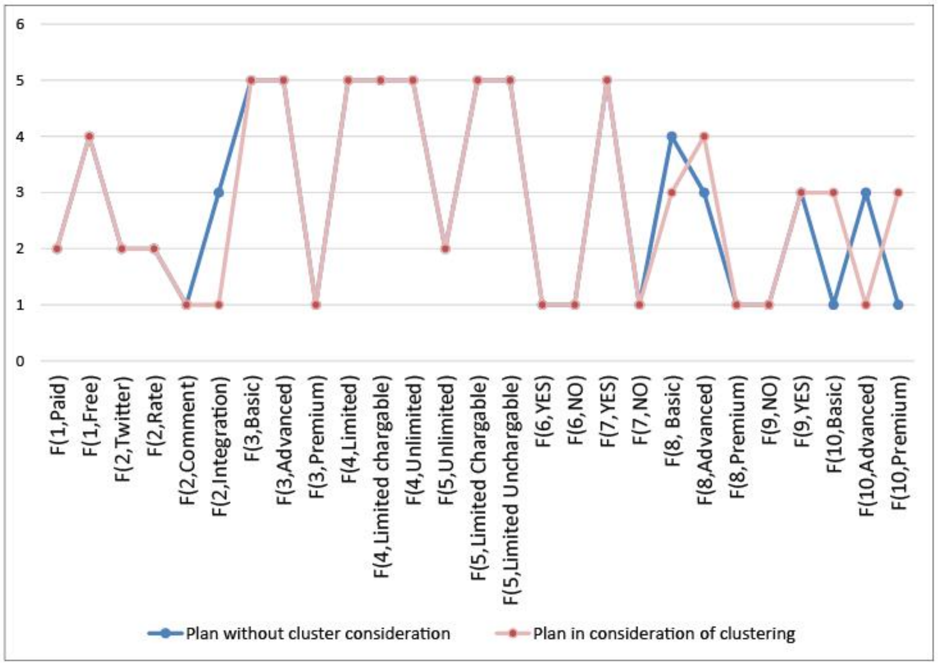
Value in consideration of clustering

Figure 14. Comparison of planning without clustering and by considering 6 clusters created from the crowd. The placements of bold points on horizontal lines showing the release quarter for each feature. The degree of optimality for five alternative release plans are compared as well (below)

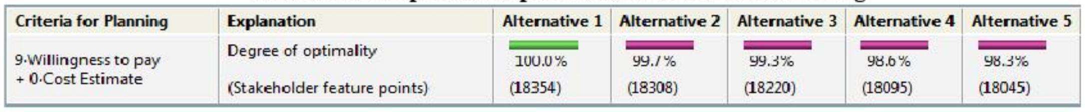
The results of re-planning based on clusters are compared in Figure 14 and Figure 15. In Figure 14 we used the 6 clusters defined in Figure 13, and we re-planning for releases. The results show significant difference in structure and value and slightly better degree of optimization. In Figure 15, we re-planned for releases considering two clusters we presented in Figure 13. For confidentiality reasons the detailed data of current customers are not available and the clustering is done only base on the potential customer groups.

5.7 Conclusions and discussion of results

The case study is done in the domain of a real industrial project for over the top TV (OTT) product and its related features. This case is mainly derived by the crowdsourcing technique and morphological analysis (see Section 4.2.2) among analytical metrics. During planning the stakeholders are involved for defining the problem constraint with the focus on cost estimation while two groups of users (i) current users (ii) potential users are involved in stating their willingness to pay for features in the crowdsourcing process. The purpose of the case study is primarily to serve as a proof-of-concept. The results are considered preliminary and no claims on external validity can be made yet.

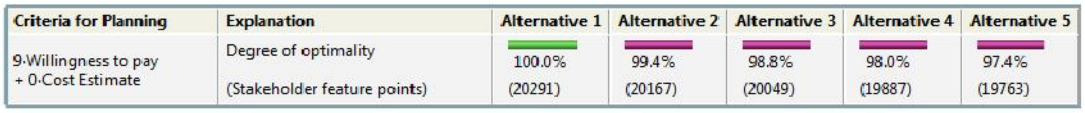


Structural comparison of plans with and without clustering



Criteria for Planning	Explanation	Alternative 1	Alternative 2	Alternative 3	Alternative 4	Alternative 5
9·Willingness to pay + 0·Cost Estimate	Degree of optimality (Stakeholder feature points)	100.0% (18354)	99.7% (18308)	99.3% (18220)	98.6% (18095)	98.3% (18045)

Value without cluster consideration



Criteria for Planning	Explanation	Alternative 1	Alternative 2	Alternative 3	Alternative 4	Alternative 5
9·Willingness to pay + 0·Cost Estimate	Degree of optimality (Stakeholder feature points)	100.0% (20291)	99.4% (20167)	98.8% (20049)	98.0% (19887)	97.4% (19763)

Value in consideration of clustering

Figure 15. Comparison of planning without clustering and by considering two clusters

As the non-trivial part of the release planning problem, the extraction and consideration of feature dependencies and synergies are highly desirable. This is supported in the AOI process the effect of synergy and dependency detection were investigated and the results showed a significant improvement in both release value (in terms of stakeholder's feature points) and the structure of the release plan.

Based on the existing real-world data, additional synthetic data was used to perform re-planning before each release (quarter of the year). This re-planning process not only changed the structure of the plan but also demonstrated the likely improvement in terms of the release value of the release as shown in Figure 12.

The crowdsourced data has wide variety and needs precise analysis to consider the exceptional group of customers as well as majority trend toward product needs. The clustering approach introduced in Section x.4.2.1 is applied to the crowdsourced data and the significant improvement in plan value were observed[1].

The relation and synergy between features are extracted with stakeholder participation and crowdsourcing, although the process of information extraction were done manually and thus may have human errors, though text mining support would be further investigated.

6. Summary and Future Research

Open innovation is seeking for significant value by looking beyond organizational boundaries and extending the scope of research and development limits to include outside institutions. The AOI approach is looking for variety of data containers to achieve the insight full and precise needs of the markets. The AOI domains of data gathering techniques are shown in Figure 16.

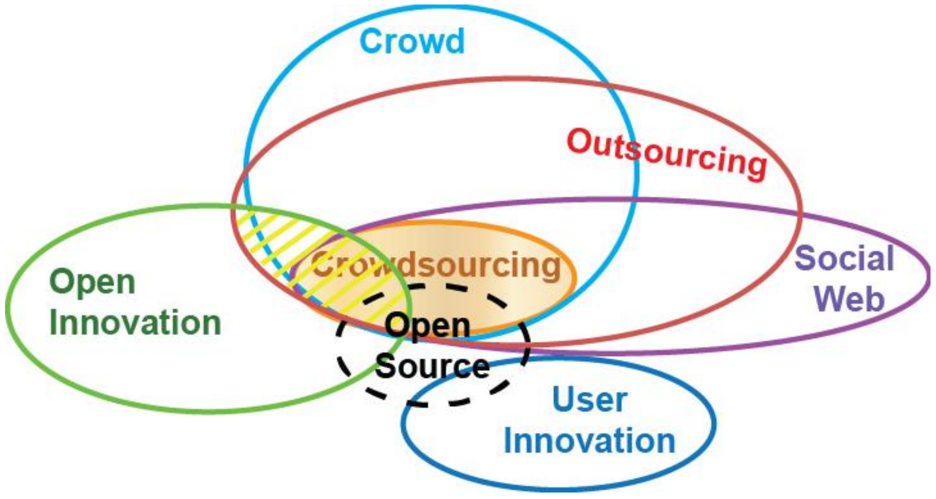


Figure 16. Open innovation related concepts - AOI use of open innovation is positioned with yellow texture

Crowdsourcing is taking advantage of web technologies and social media with the ability to actively involve users and control the community to achieve organizational tasks. This can be conceptualize as a

[1] The crowd sourced data used in this case study is available on http://ucalgary.ca/mnayebi/files/mnayebi/dataset-book-chapter.pdf

type of sourcing model which is the intersection of outsourcing and sophisticated web technologies [54]. Crowdsourcing also seen as the "small scale outsourcing" [55]. Outsourcing is a mean of contracting internal business needs or functions to outside business providers [56] which are closely connected with an institution. Open source with the spirit of peer production does not have a clear boundary between problem 'solvers' and solution 'seekers' also there is not a clear hierarchical structure of ownership and control [4].

Analytics under the premise of diversity of data implies the need for diversity and combination of analytical techniques. In this chapter, we have outlined an approach that is intended to support this process. Analytical Open Innovation is designed as an extendable framework combining the strengths of various individual methods and techniques. Combined with the idea of using broader range of internal and external knowledge containers, the AOI framework is intended to overcome current limitations of (big) data analytics

In this chapter, we have described the idea of AOI and have applied it to the problem of (analytical) product release planning. The concrete implementation called AOI@RP comprises analytical techniques used in the different stages of the release planning process. Although still in its infancy, the AOI@RP approach and its related techniques show promising results [31, 57]. From the illustrative case study, we have demonstrated that the main contributions of the AOI@RP approach are towards:

- Structuring the problem: By detecting problem dimensions.
- Defining solution space: By detecting and maintaining probable ranges of solutions and relations.
- Prediction and estimation of planning attributes by exploring the historical data for extracting features and predicting the value.
- From an enlarged and improved input for data and information of planning, we have a better chance to address the right problem. From applying advance optimization techniques, we are able to include all types of dependencies between features as well as cost and value synergies.

As outlined by Menzies [58], we are currently in the era of decision systems and leading into the discussion systems which needs to satisfy four layers of social reasoning namely *do* (prediction and decision), *say* (summarization, plan and describe), *reflect* (Trade-offs, envelopes, diagnosis, monitoring), *share* and *scale*. Our AOI platform not only supports the decision systems but also is aligned with these dimensions. Morphological analysis discussed in Section x.4.2.2 provides a unique opportunity to maintain the relationships between well-defined dimensions of the problem and to reduce the threat of wickedness in the nature of planning problem although the approach needs to be further investigated. In particular, we are planning and suggesting future research in the following four main directions:

(i) Mining inter organizational repositories will provide context specific data which helps in solving planning aspects by analogy. Previous work on automating feature analysis from distributed repositories [59] opens new way to utilize the available data on internet. Mining open forums to gather software related information would be a next step to be integrated into AOI@RP.

(ii) The new data analytics approach should be able to handle the dynamically changing data and to adjust the target of data analytics and find satisfactory solutions to the problem. Furthermore in order to create real time value while planning, AOI needs to swiftly process the dynamic data which is changing over time. Swarm intelligence as a collective behavior of decentralized, self-organized systems [60] studies the collective behavior of systems composed of many individuals interacting locally with the environment and with each other. Swarms inherently use forms of decentralized control and self-organization to achieve their goals. Swarm intelligence has been widely applied to solve stationary and dynamical optimization problems specifically in presence of a wide range of uncertainties [61].

Generally, there are two kinds of approaches that apply swarm intelligence as data mining techniques [62].

1- Techniques where individuals of a swarm move through a solution space and search for solution of the data mining task. This approach is applied to optimize the data mining techniques in order to perform effective search.

2- Swarms move data instances that are placed on a low-dimensional feature space in order to come to a suitable clustering or low-dimensional mapping solution of the data to organize the data.

As the next step we will investigate on the applicability of different swarm optimization algorithms such as particle swarm optimization [63], ant colony optimization (ACO) and brain storm optimization algorithm [64] on AOI approach to make the data analysis more efficient.

(iii) Smooth integration of the different components integrated under the AOI@RP platform label. This includes a higher degree of automation of the analysis steps to be done and the implementation of data exchange processes

(iv) More comprehensive empirical analysis is needed for the implemented framework to demonstrate its usefulness. AOI@RP is planned to perform continues evaluation with the intention to further qualify the decision-making by performing scenario-playing and other form of variation of project parameters. Also, the scalability of the whole approach needs to be investigated in terms of its effectiveness and efficiency.

Acknowledgements

This research was partially supported by the Natural Sciences and Engineering Research Council of Canada, NSERC Discovery Grant 250343-12. We very much appreciate all the comments provided by reviewers, they helped us in presenting the content in a more understandable way. We also thank Kornelia Streb for her help in editing the chapter and for preparation of high quality figures. Finally, we would like to thank the book editors for their initiative, comments provided, and the overall support in publishing this book.

References

[1] C. Ebert and S. Brinkkemper, "Software Product Management–An Industry Evaluation," *Journal of Systems and Software,* 2014.

[2] J. Highsmith, *Adaptive software development: a collaborative approach to managing complex systems*: Addison-Wesley, 2013.

[3] H. Chesbrough, *Open Innovation: The New Imperative for Creating and Profiting from Technology*. Boston, Massachusetts: Harvard Business Press, 2003.

[4] F. C. Marjanovic S, Joanna C, "Crowdsourcing based business models: In search of evidence for innovation 2.0," *Science and Public Policy,* vol. 39, pp. 318-332, 2012.

[5] T. Z. Raymond P.L. Buse, "Information Needs for Software Development Analytics," presented at the Software Engineering (ICSE), 2012 34th International Conference on, 2012.

[6] B. Vasilescu, A. Serebrenik, and T. Mens, "A historical dataset of software engineering conferences," in *Proceedings of the Tenth International Workshop on Mining Software Repositories*, 2013, pp. 373-376.

[7] P. M. Johnson, "Searching under the Streetlight for Useful Software Analytics," *IEEE Software,* vol. 30, pp. 57-63, 2013.

[8] S. Demeyer, A. Murgia, K. Wyckmans, and A. Lamkanfi, "Happy birthday! a trend analysis on past MSR papers," in *Proceedings of the Tenth International Workshop on Mining Software Repositories*, 2013, pp. 353-362.

[9] A. Hassan, "Software Analytics: Going beyond Developers," *IEEE Software,* vol. 30, p. 53, 2013.

[10] T. Menzies and T. Zimmermann, "Software Analytics: So What?," *IEEE Software,* vol. 30, pp. 31-37, 2013.

[11] H. Hemmati, S. Nadi, O. Baysal, O. Kononenko, W. Wang, R. Holmes*, et al.*, "The MSR Cookbook Mining a Decade of Research," in *MSR '13* San Fransisco, CA, 2013, pp. 343-352.

[12] G. Ruhe, *Product Release Planning: Methods, Tools and Applications*: CRC Press, 2010.

[13] N. Agarwal, R. Karimpour, and G. Ruhe, "Theme-based Product Release Planning: An Analytical Approach," presented at the HICSS-47, Hawaii, 2014.

[14] D. Port and J. Wilf, "The Value of Certifying Software Release Readiness: An Exploratory Study of Certification for a Critical System at JPL," in *Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on*, 2013, pp. 373-382.

[15] R. Berntsson Svensson, P. Lindberg Parker, and B. Regnell, "A prototype tool for QUPER to support release planning of quality requirements," in *Fifth International Workshop on Software Product Management*, 2011, pp. 57-66.

[16] B. Regnell, R. B. Svensson, and T. Olsson, "Supporting roadmapping of quality requirements," *IEEE Software,* vol. 25, pp. 42-47, 2008.

[17] E. Alba and J. Francisco Chicano, "Software project management with GAs," *Information Sciences,* vol. 177, pp. 2380-2401, 2007.

[18] B. Boehm and R. Valerdi, "Impact of software resource estimation research on practice: a preliminary report on achievements, synergies, and challenges," in *Software Engineering (ICSE), 2011 33rd International Conference on*, 2011, pp. 1057-1065.

[19] T. Klinger, P. Tarr, P. Wagstrom, and C. Williams, "An enterprise perspective on technical debt," in *Proceedings of the 2nd Workshop on Managing Technical Debt*, 2011, pp. 35-38.

[20] M. Schubanz, A. Pleuss, L. Pradhan, G. Botterweck, and A. K. Thurimella, "Model-driven planning and monitoring of long-term software product line evolution," in *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*, 2013, p. 18.

[21] G. Zorn-Pauli, B. Paech, T. Beck, H. Karey, and G. Ruhe, "Analyzing an Industrial Strategic Release Planning Process–A Case Study at Roche Diagnostics," in *Requirements Engineering: Foundation for Software Quality*, ed: Springer, 2013, pp. 269-284.

[22] V. Heikkilae, A. Jadallah, K. Rautiainen, and G. Ruhe, "Rigorous support for flexible planning of product releases - A stakeholder-centric approach and its initial evaluation," in *HICSS*, Hawaii, 2010.

[23] P. Kapur, A. Ngo-The, G. Ruhe, and A. Smith, "Optimized staffing for product releases and its application at Chartwell Technology," *Journal of Software Maintenance and Evolution,* vol. 20, pp. 365-386, 2008.

[24] P. Bhawnani, G. Ruhe, F. Kudorfer, and L. Meyer, "Intelligent Decision Support for Road Mapping - A Technology Transfer Case Study with Siemens Corporate Technology," *Workshop on Technology Transfer in Software Engineering,* pp. 35-40, 2006.

[25] J. Momoh and G. Ruhe, "Release planning process improvement - An industrial case study," *Software Process Improvement and Practice,* vol. 11, pp. 295-307, 2006.

[26] M. Lindgren, R. Land, C. NorstroÌˆm, and A. Wall, "Towards a capability model for the software release planning process - Based on a multiple industrial case study," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008, pp. 117-132.

[27] B. Regnell and S. Brinkkemper, "Market-Driven Requirements Engineering for Software Products," in *Engineering and Managing Software Requirements*, A. Aurum and C. Wohlin, Eds., ed: Springer, 2005, pp. 287-308.

[28] M. Khurum, T. Gorschek, and M. Wilson, "The software value map—an exhaustive collection of value aspects for the development of software intensive products," *Journal of Software: Evolution and Process,* vol. 25, pp. 711-741, 2013.

[29] P. Carlshamre, "An Industrial Survey of Requirements Interdependencies in Software Product Release Planning," *Fifth International Symposium on Requirements Engineering (RE'01),* pp. 84-91, 2001.

[30] Å. G. Dahlstedt and A. Persson, "Requirements interdependencies: state of the art and future challenges," in *Engineering and managing software requirements*, ed: Springer, 2005, pp. 95-116.

[31] M. Nayebi and G. Ruhe, "An Open Innovation Approach in Support of Product Release Decisions," in *Accepted ICSE 2014 - CHASE workshop*, ed. Hydrabad, India, 2014.

[32] H. Sharp, A. Finkelstein, and G. Galal, "Stakeholder Identification in the Requirements Engineering Process," in *10th International Workshop on Database and Expert Systems Applications*, Florence, Italy, 1999, pp. 387-391.

[33] T. Gorschek, S. Fricker, K. Palm, and S. Kunsman, "A lightweight innovation process for software-intensive product development," *IEEE software,* vol. 27, pp. 37-45, 2010.

[34] J. Cleland-Huang, Horatiu Dumitru, Chuan Duan, and Carlos Castro-Herrera, "Automated Support for Managing Feature Requests in Open Forums," *Communications of the ACM,* vol. 52, pp. 68-74, 1 October 2009.

[35] P. Fitsilis, V. Gerogiannis, L. Anthopoulos, and I. K. Savvas, "Supporting the requirements prioritization process using social network analysis techniques," in *Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE), 2010 19th IEEE International Workshop on*, 2010, pp. 110-115.

[36] S. M. Shahnewaz and G. Ruhe, "RELREA - An Analytical Approach for Evaluating Release Readiness," ed. Vancouver, Canada: Submitted to SEKE 2014, 2014.

[37] P. Berander and A. Andrews, "Requirements prioritization," *Engineering and Managing Software Requirements,* pp. 69-94, 2005.

[38] *International Software Product Management Association*. Available: http://ispma.org/glossary/

[39] M. Shepperd, "Software project economics: a roadmap," in *Future of Software Engineering, 2007. FOSE'07*, 2007, pp. 304-315.

[40] A. Van der Hoek, R. S. Hall, D. Heimbigner, and A. L. Wolf, "Software release management," *Proc. Sixth European Software Engineering Conference,* pp. 159-175, 1997.

[41] *What is Big Data Analytics?* Available: http://www-01.ibm.com/software/data/infosphere/hadoop/what-is-big-data-analytics.html

[42] MTurk. (2013). *MTurk*. Available: https://www.mturk.com/mturk/

[43] http://edi.lite.verybestchoice.com:3000/. *Very Best Choice light, Expert Decisions Inc.*

[44] www.releaseplanner.com. (March 2008). *ReleasePlanner®, www.releaseplanner.com (1.7 ed.)*.

[45] D. Zhang, S. Han, Y. Dang, J. Lou, H. Zhang, and T. Xie, "Software Analytics in Practice," *IEEE Software,* vol. 30, 2012.

[46] M. I. Ullah, G. Ruhe, and V. Garousi, "Decision support for moving from a single product to a product portfolio in evolving software systems," *Journal of Systems and Software,* vol. 83, pp. 2496-2512, 2010.

[47] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, 1996, pp. 226-231.

[48] T. Ritchey, "Wicked Problems--Social Messes: Decision Support Modelling with Morphological Analysis.," *Springer* vol. 17, 2011.

[49] T. Ritchey, "Problem structuring using computer-aided morphological analysis," *Journal of the Operational Research Society,* vol. 57, pp. 792-801, 2006.

[50] T. Ritchey, M. Stenström, and H. Eriksson, "Using Morphological Analysis for evaluating Preparedness for Accidents Involving Hazardous Materials," in *Proceedings of the 4th LACDE Conference*, Shanghai, 2002.

[51] A. Ngo-The and G. Ruhe, "A systematic approach for solving the wicked problem of software release planning," *Soft Computing,* vol. 12, pp. 95-108, 2008.

[52] M.-J. Montpetit, N. Klym, and T. Mirlacher, "The future of IPTV (connected, mobile, personal and social)," *Multimedia Tools and Applications* vol. 53, pp. 519-532, 2011.

[53] J. De Boever and D. De Grooff, "Peer-to-Peer Content Distribution and Over-The-Top TV: An Analysis of Value Networks," in *Handbook of Peer-to-Peer Networking*, ed USA: Springer, 2010, pp. 961 - 983.

[54] G. D. Saxton, O. Oh, and R. Kishore, "Rules of Crowdsourcing: Models, Issues, and Systems of Control," *Information Systems Management,* vol. 30, pp. 2-20, 2013.

[55] D. Gefen and E. Carmel, "Is the world really flat? A look at offshoring at an online programming marketplace," *MIS quarterly,* pp. 367-384, 2008.

[56] R. Kishore, H. R. Rao, K. Nam, S. Rajagopalan, and A. Chaudhury, "A relationship perspective on IT outsourcing," *Communications of the ACM,* vol. 46, pp. 86-92, 2003.

[57] M. Nayebi and G. Ruhe, "Analytical Open Innovation for Value-Optimized Service Portfolio Planning " in *Accepted to ICSOB conference* ed. Paphos, Cyprus, 2014.

[58] T. Menzies, "Beyond data mining; towards idea engineering," in *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*, 2013, p. 11.

[59] H. Dumitru, Gibiec, M., Hariri, N., Cleland-Huang, J., Mobasher, B., Castro-Herrera, C., & Mirakhorli, M, "On-demand feature recommendations derived from mining public product descriptions," presented at the 33rd International Conference on Software Engineering, ICSE 2011, Waikiki, Honolulu, HI; United States, 2011.

[60] E. A. Mishra, M. Das, and T. Panda, "Swarm Intelligence Optimization: Editorial Survey," *International Journal of Emerging Technology and Advanced Engineering,* vol. 3, 2013.

[61] M. Dorigo and M. Birattari, "Swarm intelligence," *Scholarpedia,* vol. 2, p. 1462, 2007.

[62] D. Martens, B. Baesens, and T. Fawcett, "Editorial survey: swarm intelligence for data mining," *Machine Learning,* vol. 82, pp. 1-42, 2011.

[63] R. Kazman, M. Klein, M. Barbacci, T. Longstaff, L. H., and J. Carriere, "The Architecture Tradeoff Analysis Method," in *International Conference on Engineering of Complex Computer Systems (ICECCS98)*, Monterey, California, 1998, pp. 68-78.

[64] Y. Shi, "Brain storm optimization algorithm," in *Advances in Swarm Intelligence,* ed: Springer, 2011, pp. 303-309.

[65] I. Hronszky and K. Kovács, "Interactive value production through Living Labs," *Acta Polytechnica Hungarica,* vol. 10, pp. 89-108, 2013.

Appendix I. Feature dependency constraints

Definition. The set CFD of coupling dependencies is presented by set of coupled features [65] based on the definition:

$x(n_1,l_1) = x(n_2,l_2)$ for all pairs of coupled features $F(n_1,l_1), F(n_2,l_2)) \in CFD$

Definition. The set PFD of precedence dependencies is defined by:

$x(n1,l1) \leq x(n2,l2)$ for all pairs of precedence features $F(n1,l1), F(n2,l2)) \in PFD$

Certain features and their related instances are not compatible with each other and do not make sense to be offered in conjunction. Detection of these incompatibilities is a complex problem itself, and we will later use MA/CCA analysis to find them.

Definition. NAND indicates that:

$x(n_1,l_1)$ NAND $x(n_2,l_2)$ if and only if features $F(n_1,l_1), F(n_2,l_2))$ cannot be offered both in conjunction

Definition. The combination of different features is creating less cost than the individual feature values. We call them the set of *cost synergies FCS*:

If ItemSet = $\{F(n_1,l_1), \dots ,F(n_y,l_y)\} \subseteq$ FCS then: Sum cost of ItemSet is decreased by Factor% if none of these items is postponed.

Similarly, from a value perspective, the combination of certain features will increase the attractiveness to the user which is called value synergy.

Definition. The set of *value synergies FVS* is defined as

If ItemSet = $\{F(n_1,l_1), \dots ,F(n_y,l_y)\} \subseteq$ FVS then: Sum value of ItemSet is increased by Factor% if none of these items is postponed.