2009

A Profit-Maximizing Method for the Partitioning of Embedded Software Features in Motor Vehicles

Oliver Baecker
University of St. Gallen, oliver.becker@sap.com

Harald Weppner
SAP Research Palo Alto, harald.weppner@sap.com

Jochen Strube
Simon-Kucher and Partners, jochen.strube@simon-kucher.com

A Profit-Maximizing Method for the Partitioning of Embedded Software Features in Motor Vehicles

Oliver Baecker
SAP Research St. Gallen, University of St. Gallen
oliver.baecker@{sap.com,unisg.ch}

Harald Weppner
SAP Research Palo Alto
harald.weppner@sap.com

Jochen Strube
Simon-Kucher & Partners
jochen.strube@simon-kucher.com

ABSTRACT

As the system design of in-car embedded systems becomes more and more modular and motor vehicles get increasingly connected to enterprise systems based on Car-2-X technology, the integration of additional embedded software features becomes technically feasible throughout the product lifecycle. For car manufacturers, this opens up the opportunity to sell additional embedded software features to their customers at a later time, thus generating subsequent revenue in addition to the initial sale. However, due to the competitive environment and customer preferences, it is impossible to apply this concept to the complete feature set. In order to support the decision, which features should be included in a shipped product and which features should be retained to generate subsequent revenue, we propose a profit-maximizing method that identifies two complementary feature bundles. To illustrate our approach, we present a numerical example, which illustrates the partitioning of embedded software features in motor vehicles.

Keywords

Embedded systems, C2X, feature partitioning, combinatorial optimization, knapsack problem.

INTRODUCTION

In recent years, the share of embedded software as opposed to hardware and mechanical parts grew significantly in motor vehicles. At the same time, the system design became more and more modular and motor vehicles got increasingly connected to enterprise systems based on Car-2-X (C2X) technology (Broy, 2006). While so far, the embedded software deployed in motor vehicles was usually altered during repair shop visits, it now becomes technically feasible to modify on-board embedded software anytime and anywhere. A modification of the embedded software configuration can be enabled for example by transferring additional software binaries to the motor vehicle or it can be based on license keys that activate additional pre-installed functionality. While the modification of the embedded software deployed in a motor vehicle is technically possible, for the time being car manufacturers are likely to limit the range of possible modifications to non safety-critical software features in the area of infotainment, navigation, or business applications. The reasons are safety concerns when it comes to the configuration of in-car embedded software.

From a business perspective, the technical feasibility of remote software modifications allows extending the functional range of on-board software at reduced costs and unlocks new revenue potential. In order to unlock this potential, original equipment manufacturers (OEMs) need to logically separate the total functional capabilities of embedded software deployed on a motor vehicle into separate modules or "features". Based on the logical separation into distinct "features" it becomes possible to sell selected features at a later point in time; a process denoted as "feature upselling". Besides the effect in terms of costs and revenue potential, the freedom to release software-based features at a later point in time has an impact on the time-to-market, since premature features do not hinder a successful market launch. When it comes to the bundling of software features during the product design phase, the challenge for OEMs is to decide which embedded software features should be pre-installed in a motor vehicle and which features should be sold separately as part of a feature upselling. While companies aim at maximizing their profit from feature upselling processes, they also need to take into account side conditions like the competitive landscape, customer needs, or market conditions. Against this background, we argue that methods of combinatorial optimization can be applied to formalize the outlined practical problem (Papadimitriou and Steiglitz, 1998). Hence, we propose a profit-maximizing method for the partitioning of embedded software features in motor vehicles. The

contribution of this paper is a method that determines two distinct feature bundles: one which contains so-called "feature upselling candidates" and one that includes all features that are pre-installed in a motor vehicle. We formalize the partitioning problem using a 0-1 Knapsack problem, which can be solved using the "Branch and Bound" algorithm (Narendra and Fukunaga, 1977). In order to demonstrate the feasibility of the proposed method, we provide a numerical example and conclude our paper with a summary and an outlook on future work.

RELATED WORK

Related work from industry and academia can be separated in two main groups. The first group concerns itself with the management of embedded software along the product lifecycle. This includes compatibility issues between differing software versions and migration paths from one software configuration to another. The second group focuses on the development of adequate Car-2-X infrastructures that enable the communication between motor vehicles and enterprise systems.

In order to identify prominent or distinctive features of software systems in a domain, Kang et al. (1990) propose a *Feature-Oriented Domain* Anal*ysis (FODA)*. The identified features are used to define the domain in terms of the mandatory, optional, or alternative characteristics of a system under evaluation. The FODA method can be applied as a preceding step of the feature bundling method proposed in this paper, because it supports the identification of relevant software features. Further research on software feature modeling can be found especially in the area of Product Line Software Engineering (PLSE) and was conducted for example by Kang et al. (2002) and Lee et al. (2002). Existing work in this area focuses on how to partition the functionality of existing embedded systems into software features. In the *Feature-Oriented Reuse Method (FORM)*, the selection of features is addressed as a part of the product development process. In contrast to the profit-maximizing selection of features discussed in this paper, the selection process described by Kang et al. (2002) is based on a product requirements analysis and does not take into account competing products. The partitioning problem discussed in this paper describes the OEMs view on feature partitioning and can be seen as a special case of a bundling problem. A car manufacturer searches for the profit-maximizing bundle of embedded software features to include in a delivered motor vehicle. While the bundling of information goods is covered for example by Bakos and Brynjolfsson (1999) and Adams and Yellen (1976), the research problem covered by this paper is different, because it does not use a dynamic price strategy and does not adjust the price of feature bundles. The European Union-funded AMPLE (Aspect-Oriented, Model-Driven Product Line Engineering) project develops concepts for an improved modularization of software variations. The research focuses on the management of software variations during the evolution of the underlying software product line and aims at enabling the (backward and forward) traceability between variations (Sousa et al., 2008). The addressed management of varying software configurations is a key enabler for feature upselling processes as discussed in this paper.

Besides research on the management of embedded software, a growing research community from academia and industry focuses on the development of the technological infrastructure that enables the integration of motor vehicles with enterprise systems. For example, the Network on Wheels (NoW) project develops a communication infrastructure for Car-to-Car and Car-2-X communication based on ad hoc principles and wireless LAN technology for road safety and business applications (Festag et al., 2008). In this context, the technologies developed for Car-2-X communication are an enabler for the feature upselling process presented in this paper. Research contributions on Car-2-X infrastructures also deal with the emergence of business applications and the like for motor vehicles (Mahfoud et al., 2008). This is relevance in the context of our research as it provides valuable input for the design of meaningful embedded software feature.

PARTITIONING OF EMBEDDED SOFTWARE

In order to be able to sell additional software features to their customers, OEMs need to design their software architecture accordingly. This means the existing embedded software share of a motor vehicle needs to be logically separated into an "embedded software platform" and n additional "embedded software features". These features can be either included in a delivered motor vehicle or they can be kept back to realize a future feature upselling. We define the two terms "embedded software platform" and "embedded software feature" as follows:

a) Embedded software platform

The embedded software platform comprises essential software components of an embedded system, which interface with its hardware components and enable more sophisticated application behavior. It consists of generic building blocks that afford further functionality and enable an embedded system to operate properly.

b) Embedded software feature

An embedded software feature is an additional software component of an embedded system, which is based on the embedded software platform. It provides more sophisticated application behavior and can be removed from an embedded

system without restraining its basic functional capability. Embedded software features can be incorporated into an existing embedded system to increase its functional range and may have dependencies with other embedded software features.

As part of the described logical separation, the share of the embedded software platform gets smaller compared to the "traditional" share of embedded software. On top of the embedded software platform, up to n additional features extend the functional range of an embedded system. Figure 1 visualizes the logical separation into the embedded software platform and n additional embedded software features.

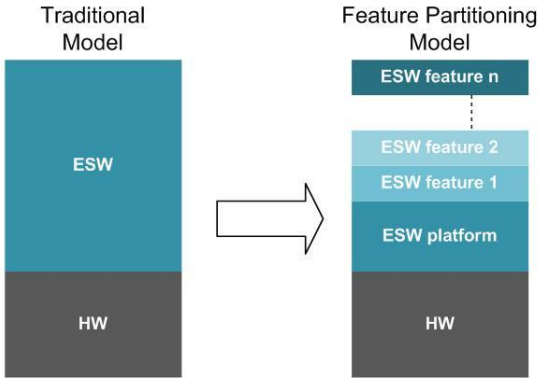

Figure 1. Logical Separation of in-car Embedded Software

In order to logically separate the embedded software platform from the embedded software features, system engineers have to decide which functionality is required for the correct operation of a motor vehicle and therefore needs to be included in the embedded software platform. In addition, they need to cooperate with sales experts to design distinct features and define their functional range. This multi-perspective approach is necessary, because a feature needs to interface with the rest of the embedded system from a technical point of view, but also needs to be appealing to customers from a sales perspective. From a customer perspective, it is also necessary that sales experts ensure that the separation of features does not become too complex. If the range of features is too large, or complex feature hierarchies occur, the customer will be overwhelmed by the feature configuration process and might choose a competing product that is easier to customize.

Based on the described separation of in-car embedded software, the OEM needs to decide which features are included in the delivered version of an embedded system and which are kept back. Each feature that is not included in this version is a candidate for a feature upselling, which extends the functional range of the embedded system and generates additional revenue for the OEM. The partitioning of embedded software features into delivered features and feature upselling candidates is visualized in Figure 2.

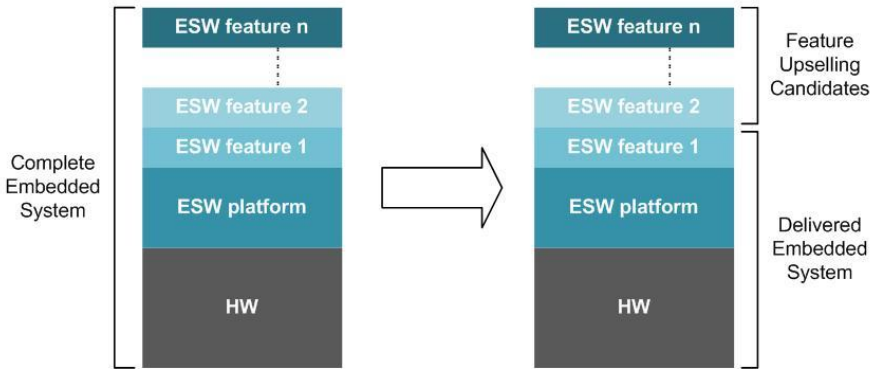

Figure 2. Partitioning of in-car Embedded Software

The decision which software features are included in the delivered version of a motor vehicle is influenced by customer demand, market conditions, the product design, and the competitive environment. From a customer perspective, functionality that is considered as commodity should not be offered as a separate feature. As an example, a car manufacturer can hardly sell on-board diagnostics like oil level readings or fuel range as separate features, since customers expect the functionality

and their willingness to pay for it is low. An example of a feature that could be sold at a later point in time during a feature upselling process is a trip reporting system for fleet management, which tracks data like mileage, GPS coordinates, and travel times and generate reports from it.

Consequently, the challenge for OEMs is to decide, which features are rolled out right away and which are envisioned to be sold later on. The feature-partitioning question thus is:

> *"Which embedded software features should be pre-installed and delivered as part of a motor vehicle and which features should be sold separately during the product lifecycle as part of a feature upselling process in order to maximize the feature-related profit?"*

The question can also be rephrased as:

> *"Which is the profit-maximizing bundle of embedded software features, to be delivered as part of a motor vehicle?"*

This latter question links to the research area of product bundling, where for example Bakos and Brynjolfsson (1999) show that an adequate bundling strategy for software features can lead to competitive advantages. A naive approach to creating a profit-maximizing feature bundle is to include no additional features in the delivered motor vehicle at all, in order to generate maximal profit from feature upsellings. However, fewer included features correlate with a reduced functional range compared to competitors. Therefore, a minimalist approach, which includes no additional features, is not feasible. A profit-maximizing strategy might be to provide as few features as possible with the initially delivered product, but still incorporate enough features to deliver a product that outpaces competitors in terms of functional range and quality. The fewer software features are included, the more profit might be generated by selling the remaining features by means of feature upsellings. The problem to be solved by OEMs is therefore to optimize the profit that is related with the sale of additional software features, while certain side conditions (e.g. the relative product quality compared to competitors) need to be satisfied. The following section provides a mathematical formalization of this decision problem based on a combinatorial optimization method.

A PROFIT-MAXIMIZING PARTITIONING METHOD

In the following section, the depicted optimization problem is formalized and a 0-1 Knapsack problem is stated, which describes the addressed decision problem (Martello and Toth, 1990). To describe the problem in a realistic way, we need to the take the corresponding profit of a feature and the probability to sell the feature into account. The profit of a feature is defined as the difference between its revenue and its costs where the revenue of a feature is determined by a company's sales division. The costs on the other hand are influenced by development costs for the respective software features, but also by the characteristics of software as a digital good (Shy, 2002, p. 182). In addition, the costs reflect how expensive it is to integrate a feature into the motor vehicle. Finally, the probability to sell a feature comes from historical sales data and empirical studies on the underlying motor vehicle market. The probability value also reflects the willingness of customers to purchase the corresponding feature for the given price. The binary variable $x_i (i = 1,...,n)$ decides whether the software feature i is included in the motor vehicle or not. If $x_i = 1$, the feature is a candidate for a future feature upselling, otherwise it is incorporated in the delivered product. The n-digit vector of decision variables $(x_1,...,x_n)$ describes which features are feature upselling candidates and which are delivered as part of the embedded software deployed on a motor vehicle. As mentioned before, the functional range, or "utility", of an in-car embedded system is a function of the set of incorporated software features. Therefore, each feature i is rated by its utility $u_i (i = 1,...,n)$. The utility of a feature is determined by customer feedback and empirical usage data. In this context, the addressed enterprise connectivity of motor vehicles in combination with a closed feedback loop between customers and OEMs allows for more accurate utility values. Over time, the utility values can be adapted to market developments and customer preferences. The sum of all utilities for incorporated features, plus a basic utility of the in-car embedded system U_{basic}, add up to the utility of an in-car embedded system $U(x_1,...,x_n)$ as depicted in the following formula:

$$U(x_1,...,x_n) = U_{basic} + \sum_{i=1}^{n} (1 - x_i) * u_i, \quad with \quad x_i \in \{0,1\}$$

Correspondingly, the utility of the own product with the complete set of features is the sum of the basic utility of the in-car embedded system plus the utility of all features. It is defined as:

$$U_{ownsystem} = U_{basic} + \sum_{i=1}^{n} u_i$$

As mentioned before, each software feature that is not included in the initially released version of a motor vehicle is a candidate for a future feature upselling. It can generate an additional revenue $r_i (i = 1,...,n)$ and has associated costs $c_i (i = 1,...,n)$. The difference leads to the profit of a feature $z_i (i = 1,...,n)$.[1] It is important to notice that $x_i = 1$ (feature i is not part of the delivered product) only implies a potential profit z_i, since the feature is not definitely sold at a later point in time. To get an estimate of the profit generated by feature upsellings, we need to take the probability that a feature i is sold into account. Therefore, we introduce the following random variable:

$$Z_i(\omega) = \begin{cases} z_i, & \omega = \textit{feature i sold} \\ 0, & \omega = \textit{feature i not sold} \end{cases}$$

The introduced random variable Z_i has the two outcomes z_i (profit for selling feature i), if the feature i is sold, and 0, if the feature is not sold. Together with corresponding probabilities of each outcome, the expected value of the random variable Z_i is calculated in a next step. It is the sum of the probability of each possible outcome multiplied by its payoff value. p_i denotes the probability that a feature i is sold during a feature upselling, while $\bar{p}_i$ is the probability that the feature is not sold. The expected value of Z_i, which is the expected profit of feature i, is therefore $E(Z_i) = z_i * p_i + 0 * \bar{p}_i = z_i * p_i$. If a feature i is a feature upselling candidate, the binary variable x_i is 1. Therefore, the expected profit is multiplied with x_i and by adding the basic profit Z_{basic} of selling an embedded system, the overall profit $Z(x_1,...,x_n)$ is calculated as:

$$Z(x_1,...,x_n) = Z_{basic} + \sum_{i=1}^{n} z_i * p_i * x_i, \ \textit{with} \ x_i \in \{0,1\}$$

The addressed decision problem can be formalized as a 0-1 Knapsack problem. The aim is to maximize the profit by choosing the right set of feature upselling candidates. As outlined before, maximizing the profit by simply including no additional features might not be feasible, because the product is in rivalry with competing products. To exceed the utility of competing products, additional features need to be included until the utility exceeds the utility of all competing products $U_j (j = 1,...,m)$. We define $U_{competitors} = \max(U_1,...,U_m)$ as the utility of the "best" competing product that needs to be exceeded. Note that every additionally included feature decreases the potential profit, because the feature cannot be sold separately at a later point in time. In the following, we use the developed functions, to formulate a 0-1 Knapsack problem. The objective function maximizes the profit related to feature upsellings. Because additional features can be sold in the future, we discount the corresponding revenue using the discount rate *d*. As a side condition, the utility of the product including the incorporated features has to be higher than the utility of competing products. We use the variables depicted in Table 1.

type	name	variable	description
decision variable		x_i	feature i is a feature upselling candidate or not ($i = 1,...,n$)
parameter	utility competitors	$U_{competitors}$	utility of "best" competing product
parameter	utility own system	$U_{own\ system}$	utility of the own product with the complete set of features
parameter	revenue feature i	r_i	additional revenue for a feature upselling of feature i ($i = 1,...,n$)
parameter	discount rate	d	used to discount future revenues in $t = 0,...,T$
parameter	costs feature i	c_i	costs of feature i with $i = 1,...,n$
parameter	probability feature i	p_i	probability of selling feature i during a feature upselling ($i = 1,...,n$)

Table 1. Variables of the Feature Partitioning Problem

[1] *To avoid confusions with the probability of selling a feature p_i, the profit of a feature is denoted z_i.*

The decision problem can be formalized as follows:

$$\textit{Maximize } F(\mathbf{x}) = \sum_{i=1}^{n}\left(\sum_{t=0}^{T}\frac{r_{it}}{(1+d)^t} - c_i\right) * p_i * x_i$$

Subject to

$$\sum_{i=1}^{n} u_i * x_i \leq U_{own\ system} - U_{competitors}$$

The difference of $U_{own\ system} - U_{competitors}$ describes how much better the own product is compared to competing products, if the complete set of features is incorporated. If this difference is negative, the model does not provide a solution. This is reasonable, because if a product that includes all features is worse than competing products, holding back features seems not to be reasonable. $U_{own\ system} - U_{competitors}$ is the window of opportunity for an OEM to nominate features for a feature upselling. $\sum_{i=1}^{n} u_i * x_i$ expresses the accumulated utility of all feature upselling candidates, which is therefore missing in the delivered product. This value cannot exceed the scope for decision-making regarding competing products. The explanatory power of the described mathematical formulation is highest, if the compared products are sold within the same price range. If the prices differ too much, it can be assumed that the utility values are also extremely different. This in turn leads to the optimal solution that a company with a superior but more expensive product should keep all of its features for a feature upselling. Given the parameters $U_{competitors}$, $U_{own\,system}$, $r_i(i=1,...,n)$, $c_i(i=1,...,n)$, $p_i(i=1,...,n)$, $u_i(i=1,...,n)$, and d, the stated 0-1 Knapsack problem can be solved using the "Branch and Bound" algorithm (Narendra and Fukunaga, 1977).

NUMERICAL EXAMPLE

A simplified numerical example that applies the presented method is given in the following paragraph. The example deals with the optimal partitioning of embedded software features of a motor vehicle. We limit the feature set to only two features and assume that costs and revenues occur in *t=0* in order to keep the example comprehensible. The two features identified during the logical feature separation are a trip reporting system (feature 1) and an infotainment system (feature 2). Data from the sales division shows that the revenue for the trip reporting system is US $ 400 (with a probability to sell it as part of a feature upselling of 0.6), while the infotainment system generates revenue of US $ 350 (with a probability to sell it as part of a feature upselling of 0.8). The costs for the trip reporting system are US $ 75 and the infotainment system costs US $ 125.

Based on customer feedback and usage-data coming from embedded in-car systems, an estimate for the utility of the trip reporting system is 23, while data shows that the utility of the infotainment system is 18. The utility of competing in-car electronic systems in the same category of automotives is estimated to be 65, while the overall utility of the company's system is 100. Using the given input data, the following 0-1 Knapsack problem can be stated:

$$\textit{Maximize } F(x_1, x_2) = (400-75) * 0.6 * x_1 + (350-125) * 0.8 * x_2$$

Subject to

$$23 * x_1 + 18 * x_2 \leq 100 - 65$$

$$x_1, x_2 \in \{0,1\}$$

The optimal solution is to include the infotainment system (feature 2) in the delivered motor vehicle and keep back the trip reporting system (feature 1) to be able to sell it during a later feature upselling. In this simplified example, the solution can be verified easily. The window of opportunity for an OEM to nominate features for a feature upselling is 100 - 65 = 35 (the utility difference between its own product including all features and the competing product). The utility of the feature upselling candidate is 23, which still undershoots the window of opportunity, while the decision to keep both features as upselling candidates would exceed it. At the same time, the trip reporting system promises a higher estimated profit of (US $ 400 – US $ 75) * 0.6 = US $ 195 compared to the (US $ 350 – US $ 125) * 0.8 = US $ 180 of the infotainment system.

CONCLUSION

In this paper, we gave an introduction to the field of feature partitioning in motor vehicles. We outlined how the increasing connectivity of motor vehicles with enterprise systems in combination with the modularization of in-car embedded systems enable the ex post integration of additional embedded software features. The work at hand describes how this trend confronts car manufacturers with the question, which embedded software features should be pre-installed and delivered as part of a motor vehicle and which features should be sold separately as part of a feature upselling process. In order to address this question, we proposed a profit-maximizing method for the partitioning of embedded software features in motor vehicles. The method compares the product under evaluation to competing products and determines the optimal feature bundle to be included in the sold product. The complementary bundle of features unlocks new revenue potential for OEMs as the included embedded software features can be potentially sold during a feature upselling. Based on the formalization of the underlying decision problem, we demonstrated the feasibility of the proposed method by means of a numerical example.

While the proposed mathematical model is a first approach to structure the feature partitioning problem, it has several limitations. Firstly, it needs to be extended in order to reflect real-world conditions more precisely. For example, the mathematical model needs to consider competing products within a different price range. Secondly, the model needs to take into account the dimension of time. This means that changing market conditions and input data need to be considered since they lead to varying profit-maximizing feature bundles over time. Thirdly, the side condition that the utility of a product including the incorporated features needs to be higher than the utility of competing products is too strict, because due to consumer heterogeneity there can also be a demand for products that are not superior to all competing products. Finally, a comprehensive model needs to reflect the interrelations and dependencies between different features.

As motor vehicles and enterprise systems, not only those operated by OEMs but also by third party software providers, become increasingly connected, feature upselling processes will become commodity. In the first place, this will concern non-safety-critical areas like navigation, infotainment, fleet management, and business applications. However, medium term, closed-loop in-car systems might become more open and allow for the integration of third-party software via dedicated and well-defined interfaces. Against this background, new revenue sources open up not only for OEMs but also for third party software providers. For OEMs, it will be crucial to decide on an optimal bundling of embedded software features. Therefore, tools and methods that support OEMs in this decision process are a key to unlock the emerging potentials.

REFERENCES

1. Adams, W. J. and Yellen, J. L. (1976) Commodity Bundling and the Burden of Monopoly, *Quarterly Journal of Economics*, 90, 3, 475-498.
2. Bakos, J. and Brynjolfsson, E. (2000) Bundling and Competition on the Internet, *Marketing Science, Special Issue on Marketing Science and the Internet*, 19, 1, 63-82.
3. Broy, M. (2006) Challenges in automotive software engineering, *Proceedings of the 28th International Conference on Software Engineering*, May 20-28, Shanghai, China, 33-42.
4. Festag, A., Noecker, G., Strassberger, M., Lübke, A., Bochow, A., Torrent-Moreno, M., Schnaufer, S., Eigner, R., Catrinescu, C. and Kunisch, J. (2008) NoW - Network on Wheels: Project Objectives, Technology and Achievements, *Proceedings of 5th International Workshop on Intelligent Transportation (WIT)*, March 18-19, Hamburg, Germany, 211-216.
5. Kang, K. C., Cohen, S., Hess, J., Novak, W. and Peterson, A. (1990) Feature-Oriented Domain Analysis (FODA) Feasibility Study, *CMU/SEI-90-TR-21*, Carnegie Mellon University.
6. Kang, K. C., Lee, J. and Donohoe, P. (2002) Feature-Oriented Product Line Engineering, *IEEE Software*, 19, 4, 58-65.
7. Lee, K., Kang, K. C. and Lee, J. (2002) Concepts and Guidelines of Feature Modeling for Product Line Software Engineering, *Proceedings of the 7th International Conference on Software Reuse: Methods, Techniques, and Tools*, April 15-19, Austin, TX, USA, 62-77.
8. Mahfoud, M., Al-Holou, N., and Baroody, R. (2008) Next Generation Vehicle Network: Web Enabled, *3rd International Conference on Information & Communication Technologies: From Theory To Applications*, April 7-11, Damascus, Syria.
9. Martello, S. and Toth, P. (1990) Knapsack problems: algorithms and computer implementations, J. Wiley, New York, NY, USA.
10. Narendra, P. M. and Fukunaga, K. (1977) A Branch and Bound Algorithm for Feature Subset Selection, *IEEE Transactions on Computers*, C-26, 9, 917-922.

11. Papadimitriou, C. H. and Steiglitz K. (1998) Combinatorial Optimization: Algorithms and Complexity, Courier Dover Publications.
12. Shy, O. (2002) The economics of network industries, Cambridge University Press, Cambridge.
13. Sousa, A., Kulesza, U., Rummler, A., Anquetil, N., Mitschke, R., Moreira, A., Amaral, V., and Araújo, J. (2008) A Model-Driven Traceability Framework to Software Product Line Development, *Proceedings of 4th European Conference on Model Driven Architecture - Traceability Workshop,* June 12, Berlin, Germany, 97-120.