A Real Time Measure of Software System Families

Sebastian Kiebusch  
Information Systems Institute  
Faculty of Economics and Management  
University of Leipzig  
Germany  
kiebusch@wifa.uni-leipzig.de

Bogdan Franczyk  
Information Systems Institute  
Faculty of Economics and Management  
University of Leipzig  
Germany  
franczyk@wifa.uni-leipzig.de

Andreas Speck  
Commercial Information Systems  
Faculty of Economics and Business Administration  
University of Jena  
Germany  
andreas.speck@uni-jena.de

Abstract

Software systems with inherent real time characteristics have become the driving force in many areas of technology like the automotive sector. Control functions of cars, driver assistance as well as systems for information and entertainment are accomplished by software driven control units. Due to the high complexity and development effort of real time systems, these resources have to be reused. Software system families are a promising solution to gain a cost reduction by reusing common software assets in different variants of an automobile. To support the economic management of this development approach we need software metrics to estimate the effort of building embedded software system families. Techniques of size measurement and cost estimation for software system families are highly insufficient in general and do not exist for the automotive domain. Therefore this article describes a conglomerate of innovative metrics to analyze a real-time perspective of embedded software system families in the automotive domain. These size metrics calculate an unadjusted measure of software driven control units to indicate and estimate their development costs.

1. Introduction

Today almost all microprocessors are implemented in embedded systems that all have to be programmed [3]. Additionally to this broad base we observe an enormous increase of real time software systems in several domains. The automotive industry is an outstanding sector to visualize this accelerated growth of technology. In these days, up to 90% of new functions and innovations in a car are enabled by embedded software technologies [3].

In contrast to this evolution, in these days not much software is reused but developed in a proprietary way for each type of car again. The huge amount of this software with many variabilities and different variants demands techniques for reuse like the concept of software system families (SSF) [3]. Within this framework a SSF is a “… set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets” [4].

To improve the possibilities of reusing software resources we develop the approach of Process Family Engineering in Service-Oriented Applications (PESOA) for the automotive and the electronic business (eBusiness) domain. The main idea of the PESOA-approach is to expand the concept of SSF by a domain independent process model for a more detailed addressing of variable and common assets.1

As a result of the rapid software inclusion in automotive control units, embedded software systems are expected to account for up to one tenth of the overall costs of a car [3]. Therefore the software development and maintenance costs are now a considerable factor for an automobile, but appropriate cost models are highly insufficient [3].

Several approaches access the areas of processes, SSF or embedded systems in the automotive domain [1], [2], [5], [6], [16]. However none of these exemplary methods measures the unadjusted size or estimates the costs of real time oriented SSF in the automotive industry. Neither the Constructive Cost Model II (COCOMO II) nor the Full Function Point (FFP) method from the Common Software Measurement International Consortium (COSMIC) considers the reuse aspects, process characteristics or external influences of a SSF development in general.

For this reason we developed the Process-Family-Points (PFP) analysis to enable a size measurement and effort estimation of embedded SSF. The PFP-Method consists of software metrics to measure the size and estimate the effort for SSF in the domains of Automotive and eBusiness. We have to mention that the latter sector

1 Further information about the PESOA-techniques are downloadable at: www.peosa.org or www.kiebusch.de
is not included in this real time focused paper but published in [9], [10] and [11]. According to this the PFP metrics of this article are a partial effort indication system to estimate the development costs of real time oriented SSF in embedded control units of automobiles.

2. Real time measurement

Real time systems differ from general information systems by the special consideration of the dimension of time [14]. An embedded real time system is always a part of a well-specified larger system, which consists of mechanical subsystems and often of a man-machine interface [13]. The control units of the automotive domain are an example application of these embedded real time systems and also the main point of the following investigations.

2.1 Categorization

No consistent classification of real time systems and their functional requirements is available in the actual technical literature. In many cases you will find a variant rich categorization of real time systems like in [13] and [14]. However the following criteria enable a general economic categorization of embedded real time systems:

- Hard timing constrains are imposed and must be validated as well as guaranteed. A missed hard timing constrain is equal to a breakdown of the well-specified larger system.
- Soft timing constrains require a less rigorous validation or guarantee and are not imposed. To miss a soft timing constrain does not affect the functionality of the well-specified larger system.

We consider real time systems in SSF. Therefore it is essential to classify the reuse of real time functions if they are a part of common or variable assets in a SSF.

Table 1: Classification of real time functions in embedded SSF

<table>
<thead>
<tr>
<th>criteria</th>
<th>asset reuse</th>
<th>functional validation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>variable</td>
<td>common</td>
</tr>
<tr>
<td>variable soft</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>(germ. EVW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>variable hard</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>(germ. EVH)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>common soft</td>
<td>-</td>
<td>X</td>
</tr>
<tr>
<td>(germ. EGW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>common hard</td>
<td>-</td>
<td>X</td>
</tr>
<tr>
<td>(germ. EGH)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1 demonstrates a classification metric in order to categorize the real time functions of embedded SSF.

Instead of other functional sizing methods this PFP categorization focuses the viewpoint of software developers according to the characteristics of the automotive domain.

2.2 Complexity weighting

Depending on the FPA and other sizing methods, the complexity of the categorized real time functions has to be weighted. Within this context the amount of input and output signals are an excellent complexity indicator of real time functions [12]. The metric in table 2 classifies the real time complexity according to the explained signal structure. All boundary values of this table are derived out of a Simulink model description of a motor control unit from DaimlerChrysler as an example of an embedded real time SSF in [15]. This tabular metric will be calibrated for a universal usage by other SSF focused real time control units.

Table 2: Complexity matrix to rate real time functions

<table>
<thead>
<tr>
<th>input</th>
<th>output</th>
<th>1 to 2</th>
<th>3 to 5</th>
<th>6 or more</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 low</td>
<td>average</td>
<td>high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>average</td>
<td>average</td>
<td>high</td>
<td></td>
</tr>
<tr>
<td>3 or more</td>
<td>high</td>
<td>high</td>
<td>high</td>
<td></td>
</tr>
</tbody>
</table>

2.3 Transformation

The categorized and complexity weighted real time functions have to be converted into the temporary size measure of unadjusted PFP. This virtual size measure is comparable to unadjusted Function Points [17], the Mark II Function Point Index [18] and the COSMIC functional size unit (Cfsu) [5].

As a matter of principle hard real time functions are more expensive to develop than soft real time functions because of validation and security aspects [12]. Beside this fact we have to identify if the real time functions are embedded in variable assets, which are less costly than their counterparts in common assets [12]. This is due to higher quality and security standards of common assets for the reason that they contain base functionalities which are reused in every product of the SSF. Additionally we have to separate among a horizontal perspective where variable real time functions are encapsulated and a vertical viewpoint where one real time function can be a part of variable as well as of common assets at the same time.

To realize a high compatibility with other functional sizing methods we assign our horizontal variable real

2 Platform for multidomain simulation and model-based design of time-varying systems.
time categories to the complexity dependant transformation factors of the FPA data perspective (5; 7; 10; 15). Within this activity we allocate the soft functions (EVW) to the lower and the hard functions (EVH) to the higher values as you can see in table 3.

Horizontal variable real time functions are reused relying on their product independent implementation frequency (IH, germ. Implementierungshäufigkeit). Hence the amount of unadjusted PFP for an EVW or an EVH is inversely proportional to their individual IH. Historical experiences in SSF development vary between different organizations. In consequence complexity dependant correction factors for variabilities (KV, germ. Korrekturfaktor- Variabilität) are required to supplement the EVW and EVH conversion factors. If the complexity repercussions are not sufficiently covered by the conversion factors, an empirical determined KV substitutes the standard KV value. A detailed derivation of the standard value and the adjustment action as well as the updating procedure is explained in [10].

Table 3: Transforming complexity weighted, horizontal variable real time functions

<table>
<thead>
<tr>
<th>Complexity</th>
<th>EVW</th>
<th>EVH</th>
</tr>
</thead>
<tbody>
<tr>
<td>low (germ. g)</td>
<td>KV&lt;sub&gt;G&lt;/sub&gt;×5/IH</td>
<td>KV&lt;sub&gt;G&lt;/sub&gt;×7/IH</td>
</tr>
<tr>
<td>average (germ. m)</td>
<td>KV&lt;sub&gt;M&lt;/sub&gt;×7/IH</td>
<td>KV&lt;sub&gt;M&lt;/sub&gt;×10/IH</td>
</tr>
<tr>
<td>high (germ. h)</td>
<td>KV&lt;sub&gt;H&lt;/sub&gt;×10/IH</td>
<td>KV&lt;sub&gt;H&lt;/sub&gt;×15/IH</td>
</tr>
</tbody>
</table>

The determination of horizontal common real time functions is basically prearranged by the transformation of horizontal variable assets. Depending on the more expensive development of common assets, they are assigned to higher conversion values which are extended about two new factors. This extrapolation is executed by equation 1 which represents the original FPA- conversion factors thru a cubic function. A detailed description of the derivation and usage of this function is included the PESOA Report No. 13/2004 [7].

\[ y = \frac{1}{6} - \frac{1}{2}x^2 + \frac{7}{3}x + 3 \] (1)

The transformation quotients in table 4 enable the final calculation of unadjusted PFP for horizontal common real time functions with consideration of complexity (low/ average/ high), validation (soft/ hard) and historical experiences in developing common assets by a correction factor (KG<sub>g/m/h</sub>). The reuse of common assets in every instance of a SSF is reflected by the amount of instantiated products (PA).

Table 4: Transforming complexity weighted, horizontal common real time functions

<table>
<thead>
<tr>
<th>Complexity</th>
<th>EGW</th>
<th>EGH</th>
</tr>
</thead>
<tbody>
<tr>
<td>low (germ. g)</td>
<td>KG&lt;sub&gt;G&lt;/sub&gt;×10/PA</td>
<td>KG&lt;sub&gt;G&lt;/sub&gt;×15/PA</td>
</tr>
<tr>
<td>average (germ. m)</td>
<td>KG&lt;sub&gt;M&lt;/sub&gt;×15/PA</td>
<td>KG&lt;sub&gt;M&lt;/sub&gt;×23/PA</td>
</tr>
<tr>
<td>high (germ. h)</td>
<td>KG&lt;sub&gt;H&lt;/sub&gt;×23/PA</td>
<td>KG&lt;sub&gt;H&lt;/sub&gt;×35/PA</td>
</tr>
</tbody>
</table>

As already mentioned, we have to convert also vertical variable real time functions. These functions are realized as a mixture of variable and common assets with a predominance of variabilities. Therefore we have to determine the quota of common signals in vertical variable real time functions like it is defined in table 5.

Table 5: Common signals in a vertical variable real time function

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Calculation</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>How high is the share of common signals in this vertical variable real time function?</td>
<td>[ G = \frac{B}{C} ]</td>
<td>[ 0 \leq G \leq \frac{1}{2} ] Closer to ( \frac{1}{2} ) means more common signals in this vertical variable function.(^3)</td>
</tr>
</tbody>
</table>

Now we have to exclude G from the variable transformation quotients in table 3 like shown in the initial part of the generic equation 2. Subsequently we have to add the product of G and the common conversion quotients from table 4 to the previous calculation according to the last part of the generic equation 2.

\[ \frac{\left(1 - G\right)\cdot KV_{g/m/h} \cdot \text{conversion factor}_{var}}{IH} + \frac{G \cdot KG_{g/m/h} \cdot \text{conversion factor}_{com}}{PA} \] (2)

By using equation 2 to join the transformation quotients from table 3 and table 4, we are able to convert complexity weighted vertical variable real time functions into the size measure of unadjusted PFP.

The transformation of vertical common real time functions is prefixed by the prior procedures. Hence we have to identify the share of variable signals (V) in a vertical common real time function which is realized by table 6.

\(^3\) The maximum value of G is \( \frac{1}{2} \) because a vertical variable real time function is usually characterized by a majority of variable (C-B) and a minority of common signals (B).
### Table 6: Variable signals in a vertical common real time function

<table>
<thead>
<tr>
<th>purpose</th>
<th>calculation</th>
<th>interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>How high is the share of variable signals in this vertical common real time function?</td>
<td>[ V = \frac{B}{C} ]</td>
<td>[ 0 \leq V \leq \frac{1}{2} ] Closer to ( \frac{1}{2} ) means more variable signals in this vertical variable function.(^4)</td>
</tr>
</tbody>
</table>

Referring to the determination of \( V \) we have to separate this contingent of variable signals from the conversion quotients of table 4 and multiply them with the transformation quotients of table 3. The subsequent merging of these two terms is demonstrated in a general viewpoint by equation 3.

\[
(1-V) \frac{\text{KG}_{g/m/h} \times \text{conversion factor com}}{\text{PA}} + V \frac{\text{KV}_{g/m/h} \times \text{conversion factor var}}{\text{IH}} \] \((3)\)

The described categorization, complexity weighting and transformation of embedded real time functions enables an unadjusted functional size measurement for automotive control units. This functional measurement method takes account of asset reuse, validation effort, functional complexity and historical experiences from the perspective of a software developer. A practical validation of this PFP step as well as a comparison to the FPA is executed at DaimlerChrysler Research and Technology at the present moment. The first results are very promising because the PFP- metrics considerate the reuse of common and variable assets in SSF.

#### 3 Example

To complete and summarize the previous theoretical investigations we apply these PFP metrics to a restricted practical case study.

Figure 1 represents an embedded real time function to administer the air condition by an engine control unit. This real time function has to be transformed into the temporary size measure of unadjusted PFP.

\(4\) The maximum value of \( V \) is \( \frac{1}{2} \) because a vertical common real time function is usually characterized by a majority of common (C-B) and a minority of variable signals (B).

The functional size of this embedded vertical common, soft real time function accounts seven unadjusted PFP according to equation 3.\(^5\)

\[ \text{unadjusted PFP} = \left( (1-V) \times \frac{\text{KG}_{g/m/h} \times 23}{\text{PA}} \right) + \left( V \times \frac{\text{KV}_{g/m/h} \times 10}{\text{IH}} \right) \]

\[ = (1-0.25) \times \frac{1 \times 23}{3} + 0.25 \times \frac{1 \times 10}{2} = 7 \]

**4**

**Figure 1:** Exemplary real time function from an embedded SSF

Beside this diagram we need the following basic information to use the discussed PFP micro analysis in this measurement situation:
- Soft timing constrains;
- Vertical variability with two variable (highlighted gray) and six common signals;
- High complexity in dependence on the signal structure and table 2;
- Quota of variable elements is \( V = \frac{2}{8} = 0.25 \) according to table 6;
- Standardized correction factors (\( \text{KG}_{g/m/h} = 1 \); \( \text{KV}_{g/m/h} = 1 \)) are used because of unavailable historical experiences;
- Variable elements are present in two instances of a SSF including tree products.

The functional size of this embedded vertical common, soft real time function accounts seven unadjusted PFP according to equation 3.\(^5\)

This measure describes the partial size of a real time function in an embedded control unit and can be used as a coarse effort indicator for a SSF development [8].

#### 4 Conclusion

The metrics in this article are independent from the development techniques of process orientated SSF and accomplish an unadjusted size measurement for real time control units in the automotive domain.

To establish an ISO/IEC 14143 compatible measurement system was not the goal of our PFP approach. This is justified in that we need to analyze the viewpoint of a software developer instead of the consumer perspective in the automotive sector. Furthermore we appreciate the flexibility to extend our approach with a PFP macro analysis which is not compliant to the generic rules of functional sizing in ISO/IEC 14143 but helpful for accurate cost estimation.
Figure 2: The PFP approach to estimate the effort in process oriented SSF

Figure 2 illustrates the entire PFP concept to estimate the effort for process oriented SSF in multiple domains. The reason that the PFP analysis focuses the eBusiness as well as the automotive sector is that both domains are characterized by processes. Therefore a process orientation increases the efficiency in a domain specific SSF development and reduces the costs of software products in these exemplary sectors.

Within the first PFP step we identify the type of count which can be a development or a reuse project as well as a count for a single product of the SSF [9]. The following PFP action which is illustrated in figure 2 defines the application boundary and the counting scope to demarcate the process oriented SSF [11].

An unadjusted size measure is calculated by the eBusiness oriented PFP micro analysis in figure 2. Within this framework we categorize, weight and transform a data as well as a process oriented view. In [9], [10] and [11] you will find a detailed description of these PFP parts.

The real time oriented PFP modules (categorization, complexity weighting, transformation) with a grey emphasis are elucidated in this article and determine an partial unadjusted size measure for embedded SSF in the automotive sector.

A completion of the domain specific PFP micro analysis is to be gained by the PFP macro analysis which considers external influences on development costs with soft characteristics. The PFP macro analysis allows a flexible modification and extension of all influencing factors. Therefore this part of the PFP analysis is versatile applicable and could be used as a potential substitution of the FPA or Mark II adjustment procedures.

Inside of the PFP macro analysis we examine twenty domain independent influences according to the approaches of FPA, Mark II and in allusion to the Object/Data Point Method. Subsequently we look at fifteen domain specific system characteristic for the eBusiness as well as for the automotive domain. After these obligatory steps it is possible to calculate adjusted PFP as a size measure which is comparable to other adjusted functional size measures. Alternatively we can determine quality adjusted PFP by regarding quality aspects in dependence on ISO/IEC 9126. With this action we loose our compatibility to FPA and Mark II but realize a more precise measure for an exact effort estimation in process oriented SSF.

The procedure of estimating SSF development costs is executed before (initial calculation), in between (interim calculation) and after (final calculation) the software engineering project according to figure 2. This enables the consideration of SSF evolution between variable and common assets. Furthermore it is possible to detect additional functionality that was not specified in the requirements but identified during development.

The additional documentation process is an integral part of the PFP approach in order to retrace prior calculations and to support future effort estimations.

To validate the PFP effort estimation system we developed a PFP counting tool and apply our approach currently at a SSF oriented project with DaimlerChrysler Research and Technology and in an eBusiness project seminar at the University of Leipzig. The first results in terms of a comparison between the PFP analysis and other sizing methods such as the FPA or COSMIC FFP are very promising. The PFP approach covers the characteristics of SSF much better than the FP or COSMIC FFP analysis particularly with regard to the special aspects of reuse, processes and general system characteristics.

During the validation process we will derive regression lines for both domains in order to forecast the effort of developing process oriented SSF. With the associated two dimensional ([quality-] adjusted PFP, development effort) diagram we are able to predict software development costs after the very early project phase of asset scoping.
References