Improving Software Effort Estimation with Human-Centric Models: a comparison of UCP and iUCP accuracy

Rui Alves  
Madeira Interactive Technologies Institute  
Caminho da Penteada,  
9020-105 Funchal, Madeira, Portugal  
rui.alves@m-iti.org

Pedro Valente  
University of Madeira  
Campus da Penteada,  
9020-105 Funchal, Madeira, Portugal  
pvalente@uma.pt

Nuno Jardim Nunes  
University of Madeira  
Campus da Penteada,  
9020-105 Funchal, Madeira, Portugal  
njn@uma.pt

ABSTRACT
Bringing human-centric models into the software development lifecycle provides unique opportunities to enhance development practice. Modeling the interactive aspects of a software system ensures a better understanding of user requirements leading to improved user interface and general usage and acceptance of the system. It also provides a unique opportunity to enhance conventional software development practices, such as effort estimation, which is known to have major deviations. In this paper we illustrate this mutual benefit presenting a statistical analysis of the effort estimation for seven real world software development projects. We contrast a conventional use-case points (UCP) method with iUCP an HCI enhanced method. Here we propose an enhancement of the iUCP original effort estimation formula. This results in an improved mean deviation of iUCP over UCP supporting the claim that reflecting HCI concerns into internal SE artifacts generates more accurate estimations of software development effort. Our results provide additional evidence of the benefits of using human-centric models to enhance the software development practice, in particular for long lasting challenges like generating accurate project estimates early in the development lifecycle.

Author Keywords
Software engineering; user-centered design; effort/cost estimation; use case points; interactive use case points.

ACM Classification Keywords
D.2.9

INTRODUCTION
In recent years the software engineering (SE) and human-computer interaction (HCI) communities have tried to bridge methods and techniques that are successful for both software development and interaction design. The cross-fertilization of these disciplines is hard. Methods and techniques are developed independently and are underused largely because of a lack of common understanding between the two communities. Despite the fact that practitioners often must work together in multidisciplinary teams, examples of the lack of communication are still evident. Many mature and successful HCI techniques are unknown and unrecognized by software developers – for instance user roles and personas, human-activity modeling and contextual inquiry/design [20]. Although these techniques tackle a major problem of SE (requirements and user-involvement) they are understood by too few software practitioners and are still far from large-scale adoption.

Creating product cost estimates, early in the development lifecycle, is a challenge for the software industry. In this paper we focus on software effort estimation, one of the software development practices, which is known to have major deviations often resulting in important wastes of time and money [4, 10, 17, 19]. They require that developers agree on the concepts driving the estimations and rely on substantial data from past projects as well as constant feedback and fine-tuning. On top of failing to comply with developers’ and managers’ expectations, these inaccuracies can ultimately lead to company collapse. Therefore, the ability to estimate development effort early in the project lifecycle is critical to the SW industry [17]. However, with few exceptions [3], existing estimation methods fail to produce reliable results. Several authors reported on how estimation techniques perform when compared to real project data. Kemmer has identified huge estimation errors, in the range of 500-600% [17] while Collolpy analyzed 12 studies in which the average estimation error ranged from 13% to 413% [4]. Specifically for UCP the literature provides contradictory data. Carroll reported a widely positive study on 200 projects, conducted in one company, with less than 9% deviation in 95% of the projects [3]. On the opposite side, Crosby, in a study conducted with students, found large variations with a maximum of 500% difference between two estimates, for the same set of requirements [8]. Consequently, early estimates are a relevant challenge and an opportunity to bridge the gap between HCI and SE early in the lifecycle [20].

In this paper we explore how bringing an HCI insight to the effort estimation SE technique can improve the accuracy of the assessment. We build on previous work [22] proposing changes to the popular Use Case Points (UCP), estimation method by Karner [16]. These changes take advantage of...
the cross-fertilization of disciplines leveraging the enhanced information that can be extracted from HCI techniques - like actors, roles and essential use-cases - to improve the software estimation model. Our research with data from seven real-world projects also provides additional evidence by comparing the original UCP method with the HCI enhanced Interactive UCP (iUCP), by Nunes et al. [22].

The next section provides a brief review of the software effort estimation state of the art, as well as an assessment of the estimation accuracy reported in the literature. The following section defines the main concepts of software estimation followed by a description of the UCP and iUCP methods used in the scope of our study. We then present our research question and hypothesis, as well as the data gathering process and methodology described in our experiment design. The results section includes the raw data collected and the statistical analysis. In the discussion section we present the factors impacting our results building on this to present an improvement proposal, which was later tested by reassessing the results with the new formula. Finally we present our conclusions and the envisioned future work.

**Background and State of the art**

There are many different software estimation methods from those that rely on analogies, expert estimations and artificial intelligence techniques [17]. The UCP methods discussed in this paper are classified as parametric models and follow the long-lasting tradition of other popular methods like Function Point Analysis (FPA), proposed by Albrecht [1], and the Constructive Cost Model (COCOMO), pioneered by Boehm [2]. FPA assigns points to each function in an application, further adjusting it for environmental and technical factors like complexity, developer skills and risk. COCOMO, which is still evolving today under the sponsorship of the Center for Systems and Software Engineering at USC [4], uses statistical returns to calculate project costs and duration within a given probability. The underlying assumption of parametric models is that statistically significant historical data exists to drive the factoring of the models. However, companies struggle to find a consistent definition of functions and environmental factors across multiple projects and development platforms.

With the advent of object-oriented software engineering, use-cases emerged as the dominant technique for structuring requirements [10]. This technique, established by Jacobson, was further integrated in the Unified Modeling Language (UML) and the commercial Rational Unified Process (RUP) thus becoming the de facto standard for requirements modeling in SE. Later, Karner, also from Rational, created a software estimation technique that assigns points to use-cases in much the same way that FPA assign points to functions. This technique was named UCP and was integrated in the unified process, receiving tool support from popular UML tool vendors. The UCP model became popular due to its relative simplicity and applicability at early stages of development. UCP was practical for early estimation of software size and effort at the end of the analysis phase (requirements specification in the cone of uncertainty [2]).

In the last years several proposals to enhance UCP were published, namely the Simplified UCP by Ochodek [23] and iUCP by Nunes [22]. The simplified UCP suggests a set of simplifications to UCP, in particular: i) estimation with and without unadjusted actor weights (UAW) with similar prediction accuracy; ii) estimation of use case complexity based on steps instead of transactions; and iii) a reduction of the number of adjustment factors from 21 to 6 (two environmental factors and four technical complexity factors) maintaining accuracy [23]. Conversely iUCP provides an enhancement of UCP incorporating HCI concerns and claims to contribute to more consistent effort estimation. iUCP is based on revised actor and use-case assessment criteria emerging from enhanced human-centric models of requirements such as the ones proposed by the Wisdom method [21]. The rationale is that for interactive system development, early estimates based on models of requirements, can only be accurate if they reflect the HCI concerns related to users and their interaction with the system [20].

**Foundations**

In this section we present the core concepts used by UCP and iUCP, namely actors, roles, use cases, user intentions and system responsibilities. Since both UCP and iUCP rely on these constructs to provide the basic parametric estimation, their clear definition is important to accurately apply the methods. In fact, a major source of inaccuracy introduced by these methods comes from the different interpretation of what is a use-case or an actor. This is one of the improvements of iUCP, which bases the estimation on the HCI definitions of actor and use-case, thus providing a more consistent baseline and consequently more accurate estimations. An actor specifies a role played by a user or any other system that interacts with the subject [24]. A role constitutes a relationship between a user and a system and is defined by a set of characteristic needs, interests, expectations, behaviors and responsibilities [28]. There are many definitions of use-case and this is reportedly a problem with this widespread concept. A traditional use case is “a specific way of using the system by using some part of the functionality and constitutes a complete course of interaction that takes place between an actor and the system” [10]. On the other hand, essential use cases are “single, discrete, complete, meaningful, and well-defined task of interest to an external user in some specific role or roles in relationship to a system, comprising the user intentions and system responsibilities in the course of accomplishing that task, described in abstract, technology-free, implementation independent terms using the language of the application domain and of external users in role” [8]. As prescribed in [21] a use case can be detailed with an activity diagram, which is composed by two major sections:
user’s intentions and system responsibilities. User intentions are meaningful and complete sets of actions required to achieve a goal [21] (completion of the use case). System responsibilities are system components (objects) that make the “coordination, sequencing, transactions and control of other objects” [21].

**The UCP and iUCP Estimation Methods**

Both UCP and iUCP rely on five major components to estimate effort: unadjusted actor weight (UAW), unadjusted use case weight (UUCW), technical complexity factor (TCF), environmental complexity factor (ECF), and productivity factor (PF). iUCP differs from UCP on the complexity assessment and weighting of UAW and UUCW. The UAW is the point size of the software that accounts for the number and complexity of actors.

In the original UCP there are three actor weights [16]:

- simple (weight 1) for actors representing another system with a defined API.
- average (weight 2) for actors that interact with another system through a protocol, or a human interaction with a line terminal.
- complex (weight 3) for actors that interact through a graphical user interface.

In the revised iUCP there are six actor weights [22], which reflect the HCI understanding of the complexity of the interaction:

- simple system actors (weight 1) communicate through an API.
- average system actors (weight 2) communicate through a protocol or data store.
- simple human actors (weight 3) are supported by one user role.
- complex system actors (also a weight of 3) communicate through a complex protocol or data store.
- average human actors (weight 4) are supported by two or three user roles or one focal role.
- complex human actors (weight 5) are supported by more than three user roles or more than one focal role.

The UUCW is the point size of the software that accounts for the number and complexity of use cases and relies on the number of transactions. In UCP the transactions are the total number of activities or steps (user intentions and system responsibilities) in the use case [5]. Instead, iUCP considers transactions as the system responsibilities only. Both UCP and iUCP weight the UUCW levels equally [22]:

- simple use cases (weight 5): three or less transactions.
- average use cases (weight10): four to seven transactions.
- complex use cases (weight15): eight or more transactions.

Thus, both UCP and iUCP rely on this generic formula to calculate the estimated effort (EE), in hours:

\[
EE = UCP \times PF
\]

Where PF is the Productivity Factor (the number of hours an organization needs to implement one use case point) and the UCP is the number of points. The UCP formula is then:

\[
UCP = UUCP \times TCF \times ECF
\]

The UUCW is the Unadjusted Use Case Points, calculated by this formula:

\[
UUCW = \sum_{i=1}^{k} (n_i \times W_i)
\]

Where \(i\) is a category of actor and \(k\) is the maximum number of categories an estimation method admits. \(n_i\) is the number of items of category \(i\) and \(W_i\) is its weight.

The Unadjusted Use Case Weight (UUCW) formula is:

\[
UUCW = \sum_{i=1}^{3} (n_i \times W_i)
\]

Where \(i\) is a category of use case, \(n_i\) is the number of items of category \(i\) and \(W_i\) is the weight of category \(i\).

The TCF is the factor that is used to adjust the size based on technical considerations and the ECF is used to adjust the size based on environmental considerations. The overall effort estimation formula is:

\[
EE = (UAW + UUCW) \times TCF \times ECF \times PF
\]

Which fully expanded as follows:

\[
EE = \left( \sum_{i=1}^{k} (n_i \times W_i) + \sum_{i=1}^{3} (n_i \times W_i) \right) \times \left( C_1 + C_2 \times \sum_{i=1}^{f} (TF_i \times W_i) \right) \times \left( C_3 + C_4 \times \sum_{i=1}^{m} (EF_i \times W_i) \right) \times PF
\]

The modifications proposed by iUCP preserve the original UCP model integrity but introduce a deeper understanding of the HCI concerns captured in human-centric models of the software system under construction. The goal of iUCP is to help software developers and interaction designers to apply heuristics that are suitable for interactive applications and work consistently across and within projects [22]. In the next section we present the research question that drove our current study.

**RESEARCH QUESTION**

Combining SE and HCI provides new opportunities for collaboration between interaction designers and software developers. This helps developers to see the advantage of using HCI techniques early on. Conversely, interaction designers can better understand their models’ impact and
recognize user interface (UI) elements’ impact at the architecture level. This builds common ground for activities such as prioritizing development and planning releases.

Our research question builds on previous work where we provided evidence that iUCP produces size estimations more consistent in their assessment of use-case complexity and overall UCP unadjusted complexity. This claim was verified with reported less variance between estimations produced by iUCP when compared with UCP [22].

Here we want to test if iUCP actually produces better estimates than the conventional UCP method, grounded in the possibility that the HCI perspective introduced by iUCP will produce more accurate estimations than the original method. Hence, we have formulated the following hypothesis, $H_1$: iUCP estimation accuracy is better than UCP. Its corresponding null hypothesis, $H_0$, is that iUCP accuracy is not better than UCP. In this setting, our independent variable is the estimation method, which has two levels (UCP and iUCP) while our dependent variable is the estimation accuracy, i.e., the estimation deviation from real effort reported in each project analyzed.

### DATA GATHERING

We realized that most of the publications in this area are studies carried out in a single organization, either academic or from industry. In order to improve our contribution and reduce bias we tried to include projects from different sources in this study. Consequently, we contacted several organizations but, mainly due to confidentiality issues, data gathering proved to be a major difficulty.

Project selected for our study were required to include detailed use case models, which complied with the essential use case modeling directives (allowing the identification of the users and roles, user intentions and system responsibilities). In addition projects were required to have detailed post-mortem actual effort reports. The use-case models were required to calculate the estimates while the reported effort was used to assess the estimation deviation, by comparing the calculated estimation with the actual effort. These constraints impacted our study by reducing the sample size due to a number of rejected projects that failed to comply with the study requirements. Access to the real effort reports proved to be difficult, not only because of the underlying costs but also because several companies were not willing to disclose their confidential project data.

The study procedure included four major activities:

1. Collect project documentation.
2. Produce iUCP and UCP estimates for each project.
3. Compute the deviation from reported effort.
4. Examine the significance of the results.

In the end our study included seven projects from three organizations: Logica [18], GMV [12] and GDAI [12]. Logica is a leading multinational business and technology service company employing 41,000 people in 41 countries. The company delivers business-consulting, systems development and integration, and outsourcing across several industries. GMV is a multinational software house mainly devoted to aerospace, defense and security markets, which employs more than 1000 people worldwide. GDAI is an internal software development unit of the University of Madeira (UMa) involving a team of one coordinator and five software engineers. GDAI develops and manages the University information systems, which support all the academic activities and integrates with third parties standard business applications.

GDAI contributed with five projects (including one project previously analyzed in [27]) and both Logica and GMV provided one valid project each. Table 1 summarizes the seven projects analyzed in terms of their duration and general description. Three projects had duration of three months and the others 10, 13 and 30 months.

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Duration (months)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Facilities access control</td>
<td>3</td>
</tr>
<tr>
<td>P2</td>
<td>Billing system integration with the financial management</td>
<td>13</td>
</tr>
<tr>
<td>P3</td>
<td>Student’s accreditations request and approval</td>
<td>3</td>
</tr>
<tr>
<td>P4</td>
<td>Disciplines accreditation</td>
<td>3</td>
</tr>
<tr>
<td>P5</td>
<td>Automation of the Academic Office’s requests</td>
<td>3</td>
</tr>
<tr>
<td>P6</td>
<td>An infrastructure occurrences management tool</td>
<td>30</td>
</tr>
<tr>
<td>P7</td>
<td>A web based accounting system</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 1. Analyzed projects.

Despite the fact that our sample was small, we decided to proceed with the experiment to understand the problems impacting estimation and pave the way for future work. Next, we describe the methodology used in this study.

### METHODOLOGY

In this section we discuss how the study was conducted and identify the main problems, as well as the rational behind the decisions made in order to overcome them.

#### Experiment Design

In this study we measure the iUCP method performance against UCP by comparing the estimation results alongside the real effort reported for each project analyzed. A within-subjects design was selected, where there are no ordering effects. In order to make this comparison possible we had to comply with some trade-offs. For instance, the complexity of the uses cases assessment was based exclusively on the number of transactions, because most of the projects did not produce the robustness (architecture) model, making it impossible to consider the number of entities in each use
In order to factor out the impact of the TCF and ECF we decided to level these factors, setting them to one in both iUCP and UCP. The rational for this decision is related to the fact that we lacked data to accurately assess all the 21 factors involved in calculating the TCF and ECF factors. In addition, to avoid additional bias, the PF used was leveled to 30 hours, as experimentally suggested by Karner from the of 20-30 hours range [16]. These modifications resulted in the following effort estimation formula:

\[ EE = (\text{UAW} + \text{UUCW}) \times 1 \times 1 \times 30 \]

In summary our analysis singles out the three factors that do not change the unadjusted actor and use case estimations. This way we are able to compare the methods in what they really differ and not the implication of the technical, environmental and productivity factors. The deviation, i.e., the accuracy of each method in each project, will be calculated using the magnitude of relative error (MRE):

\[ \text{MRE} = \frac{\text{Real effort} - \text{Estimated Effort}}{\text{Real Effort}} \]

In the next section we present our study results with both the raw data and the statistical analysis conducted.

RESULTS

In order to compute the unadjusted actor and use-case weights we collected information about the models of all the projects involved in this study. That data was then used to perform the categorization of actors and use cases leading to the statistical analysis performed to test \( H_1 \).

Raw Data

We categorized actors (Table 2) and use cases (Table 3) according to each method (iUCP and UCP), as previously described. Then we calculated the UAW, UUCW and the UUCP for each project (Table 4).

Table 2 presents the actors’ raw data categorization, according to both iUCP and UCP methods. In iUCP the majority of actors (55%) are categorized as simple human actor, whereas in UCP the vast majority of actors (85%) are categorized as complex, yet leading to the same weight (3). iUCP adds an extra weight to the UAW value by categorizing six more actors (30%) as average human actors, with a weight of 4.

<table>
<thead>
<tr>
<th>W</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>3</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P6</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P7</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Actors categorization.

Table 3 presents the use cases’ raw data categorization, according to both iUCP and UCP methods. The collected data shows that iUCP tends to categorize the vast majority of use cases as simple (84%), whereas UCP tends to distribute more evenly this categorization (47% as simple, 23% as average and 30% as complex use cases). As a result UCP adds an extra weight to the UUCW, when compared to iUCP.

<table>
<thead>
<tr>
<th>w</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>5</th>
<th>10</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>4</td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>1</td>
<td>4</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td>1</td>
<td>2</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P6</td>
<td>34</td>
<td>2</td>
<td>24</td>
<td>11</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>P7</td>
<td>8</td>
<td></td>
<td>5</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Use Cases categorization.

Table 4 presents the calculated points for actors (UAW), use cases (UUCW) and the resulting UUCP. The figures confirm that in iUCP the UAW exceeds UCP in 14%. Regarding UUCW, iUCP accounted for 37% less than UCP. The resulting UUCP is always smaller for iUCP. The total number of points of iUCP is 32% less than UCP.

j) Simple Use Case, k) Average Use Case, l) Complex Use Case, w) Weight
The estimation deviations to the reported real effort are presented in Table 5. The projects real effort duration for the complete development of the requirements spans a wide range. For instance P6 is 29 times larger than P5. The estimated hours are the UUCP presented in the previous table, multiplied by the PF (30 hours). The absolute deviation column presents the deviations obtained for both iUCP and UCP.

Table 5. iUCP and UCP real effort, estimated effort and absolute deviations (MRE).

<table>
<thead>
<tr>
<th>Real Effort (Hours)</th>
<th>Estimated</th>
<th>Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>iUCP</td>
<td>UCP</td>
</tr>
<tr>
<td>P1</td>
<td>598.5</td>
<td>330</td>
</tr>
<tr>
<td>P2</td>
<td>2357.4</td>
<td>1830</td>
</tr>
<tr>
<td>P3</td>
<td>2282.0</td>
<td>900</td>
</tr>
<tr>
<td>P4</td>
<td>1134.0</td>
<td>1740</td>
</tr>
<tr>
<td>P5</td>
<td>434.0</td>
<td>990</td>
</tr>
<tr>
<td>P6</td>
<td>12889.5</td>
<td>5880</td>
</tr>
<tr>
<td>P7</td>
<td>3170.0</td>
<td>1380</td>
</tr>
</tbody>
</table>

The deviations obtained range from 0.3% to 252.5%, including three results with deviations above 100% (UCP in P4, and both iUCP and UCP in P5). The mean of deviations was 60.0% for iUCP and 72.5% for UCP. In the following section we detail our statistical analysis.

Statistical Analysis
We used SPSS 20 [11] to analyze the deviation data normality with the Skewness and Kurtosis and Kolmogorov–Smirnov tests (p<0.05). We concluded that our data is not normal. The deviations mean favored iUCP over UCP (60% versus 72.5%). Regarding data dispersion, iUCP standard deviation (32.6%) was also more consistent than UCP (89.9%), as depicted in Figure 1.

DISCUSSION
The major outcome of our statistical analysis is that we cannot reject the null hypothesis. In fact, the only result that point towards H₁ is the average deviation, in which iUCP result is better than UCP (60.0% versus 72.5%).

Based on our findings and the previous experience applying UCP and iUCP, we concluded that the results were almost unacceptable for both methods thus suggesting that other factors should have impacted the estimations. This is in line with the idea that the original UCP, largely applied in software engineering industry, should provide better results.

In fact Carroll supported this claim in a study involving more than 200 projects, in which an average deviation of...
9% was obtained in about 95% of the sample [3]. Moreover, the cone of uncertainty reports that by the end of the requirements specification phase, where UCP and iUCP are applicable, the deviation should be less than 50% [2].

As a consequence of these results, and the close analysis and discussion of our projects, we considered two different approaches to draw more conclusions from our study. Firstly, we calculated what would be the sample size needed in order to reach statistical significance. Considering that we cannot control either the effect size or the power, but admitting an increased sample with a hopefully large effect size of 0.5, the minimum required sample size would be 28 projects (maintaining a power of 0.8 and \( \alpha=0.05 \)). This possibility is far more encouraging than the 223 sample size stated in the previous section. Secondly, we inspected which factors impacted the results and how we could generate an improved proposal for effort estimation. These factors are discussed in the next section based on the available information of the software development process for each project.

Factors Impacting Results
During our study we found that deviations were influenced both by the estimation method applicability and by the real effort reporting. We also found that the estimation could be influenced by factors impacting the productivity factor, such as the requirements volatility, the software framework and the project type.

Moreover, the lack of standardization in the use case modeling techniques also impacts the estimation, because the number of transactions (system responsibilities in iUCP) and steps (the total number of activities in a use case, in UCP) has a direct impact on the final UUCP and varies significantly across organizations. For instance, iUCP estimation of average complexity use cases differed from 40% in one organization to 4.5% in another.

Furthermore, TCF and ECF are prone to bias influenced by the estimator experience [19]. Our study did not explore the estimation of these factors and hence this bias is not present in our results. However, the real effort conveyed is impacted by over-reporting, which could happen due to organizational issues, such as the financing sources, and the method used by the team to report the time spent on the project. All these issues are interdependent and complex and should be further explored. The PF significantly impacts the final effort estimation since it is multiplied by the resulting UCP (use case point). For instance, if the UUCP is 100, by applying a PF of 30, the estimation results in 3000 hours, whereas applying a PF of 15, the final estimation will change to 1500 hours. As a result of our analysis we concluded that there were three major factors impacting the estimation: 1) requirements volatility, 2) software framework, and 3) project type. This analysis led to a new proposal, that we present in the next section.

An Improvement Proposal
Requirements volatility (RV) (1) refers to the changing of requirements before the project is concluded. RV is a factor that affects negatively the productivity of a software development team. On average the RV impacts the development costs of a project by nearly 50% [9]. The original UCP method [16] addresses this with the ECF, namely in the sixth factor (T6 – Stable requirements) resulting however, in a minor impact in the final estimation. We argue that this factor is undervalued. Thus, we propose PF to normalized where a factor of 0.5 should be added to the PF of the projects affected by this situation (Table 6).

The software framework (SF) (2) refers to the development tools supporting the development process, and in particular important issues like separation of concerns leading to n-tier software architectures [25]. Taking the 3-tier architecture, we consider that if changes are made to any of the three components (database, internal logic or user-interface) during the project development, a 25% extra value should be added to the PF (i.e., 25% if changes are made in the database, 25% if changes are made in the user-interface, plus 25% if the internal logic is changed). Moreover, if a new SF is created to develop the project, a value of 50% should be considered (Table 7).

The project type (PT) (3) describes the project nature whether it is new (adds functionality) or is it maintenance (changes existing functionality). Regarding maintenance, Cote and St-Pierre reported a 20% effort reduction when dealing with modified components, 66% when dealing with suppressed components, 94% when dealing with untouched components and 0% for new components [9]. Since maintenance projects involve dealing with a mixture of the aforementioned situations, we settled for an initial value of -25% for this project type (Table 8).

### Requirements Volatility Proposed Weights

<table>
<thead>
<tr>
<th>Requirements Volatility</th>
<th>Description</th>
<th>RV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volatile</td>
<td>Requirements changed</td>
<td>0.50</td>
</tr>
<tr>
<td>Stable</td>
<td>Requirements remained stable</td>
<td>0</td>
</tr>
</tbody>
</table>

### Software Framework Proposed Weights

<table>
<thead>
<tr>
<th>Software Framework</th>
<th>Description</th>
<th>SF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Established Development Framework</td>
<td>The project is developed using an already established software framework</td>
<td>0</td>
</tr>
<tr>
<td>Improvement of the existing software framework</td>
<td>1 MVC component changed</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td>2 MVC components changed</td>
<td>0.50</td>
</tr>
<tr>
<td></td>
<td>3 MVC components changed</td>
<td>0.75</td>
</tr>
<tr>
<td>New Software Framework</td>
<td>Development of a new software architecture</td>
<td>0.50</td>
</tr>
</tbody>
</table>

Table 6.Requirements Volatility Proposed Weights.

Table 7.Development Framework Proposed Weights.
The application of the mentioned factors results in the following estimation formula, where TCF and ECF are replaced by (RV+SF+PT+1):

\[
\text{Effort} = (\text{UAW} + \text{UUCW}) \times (\text{RV} + \text{SF} + \text{PT} + 1) \times \text{PF}
\]

The experiment was repeated using the new formula, which is presented in the following section.

**Reassessing the Results with the New Proposal**

We recalculated the estimations, considering the factors that impacted each project (Table 9).

<table>
<thead>
<tr>
<th>Project</th>
<th>RV</th>
<th>SF</th>
<th>PT</th>
<th>Final PF</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>30.0</td>
</tr>
<tr>
<td>P2</td>
<td>50%</td>
<td>0%</td>
<td>-25%</td>
<td>37.5</td>
</tr>
<tr>
<td>P3</td>
<td>0%</td>
<td>50%</td>
<td>0%</td>
<td>45.0</td>
</tr>
<tr>
<td>P4</td>
<td>0%</td>
<td>25%</td>
<td>0%</td>
<td>37.5</td>
</tr>
<tr>
<td>P5</td>
<td>0%</td>
<td>0%</td>
<td>-25%</td>
<td>22.5</td>
</tr>
<tr>
<td>P6</td>
<td>50%</td>
<td>50%</td>
<td>0%</td>
<td>60.0</td>
</tr>
<tr>
<td>P7</td>
<td>50%</td>
<td>50%</td>
<td>0%</td>
<td>60.0</td>
</tr>
</tbody>
</table>

**Table 9. Corrected PF.**

Considering the new PF we calculated the new effort estimation (Table 10) and the related deviations (Table 11).

In Table 11 we compare how the new proposal affected each method’s performance. In iUCP the estimation accuracy, was better in five out of seven projects, whereas in UCP the results were worst in four out of seven.

<table>
<thead>
<tr>
<th>Project</th>
<th>Original Effort</th>
<th>Original iUCP</th>
<th>Original UCP</th>
<th>Proposal iUCP</th>
<th>Proposal UCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>598.5</td>
<td>330.0</td>
<td>600.0</td>
<td>330.0</td>
<td>600.0</td>
</tr>
<tr>
<td>P2</td>
<td>2357.4</td>
<td>1830.0</td>
<td>3090.0</td>
<td>2287.5</td>
<td>3862.5</td>
</tr>
<tr>
<td>P3</td>
<td>2282.0</td>
<td>900.0</td>
<td>2070.0</td>
<td>2250.0</td>
<td>3105.0</td>
</tr>
<tr>
<td>P4</td>
<td>1134.0</td>
<td>1740.0</td>
<td>2610.0</td>
<td>2175.0</td>
<td>3262.5</td>
</tr>
<tr>
<td>P5</td>
<td>434.0</td>
<td>990.0</td>
<td>1530.0</td>
<td>742.5</td>
<td>1147.5</td>
</tr>
<tr>
<td>P6</td>
<td>12889.5</td>
<td>5880.0</td>
<td>7530.0</td>
<td>11760.0</td>
<td>15060.0</td>
</tr>
<tr>
<td>P7</td>
<td>3170.0</td>
<td>1380.0</td>
<td>1830.0</td>
<td>2760.0</td>
<td>3660.0</td>
</tr>
</tbody>
</table>

**Table 10. Estimated effort in hours, with the original and the new formula (per project).**

1Software framework changes in two levels (MV)

2New software framework

Analyzing the new data normality trough Skewness and Kurtosis, as well as the Kolmogorov–Smirnov test, we conclude that our data is now normal (p>0.05).

The mean results favored again iUCP over UCP (33.4% versus 69.2%) - the original means were 60% and 72.5%, respectively. Regarding data dispersion, iUCP standard deviation (36.4%) was also more consistent than UCP (76.0%), as depicted in Figure 2. Again the original standard deviation values were 32.6% and 89.9%, respectively. A closer analysis of the box-plot in Figure 2 shows that the “iUCP_Proposal” and “UCP_Proposal” charts (for iUCP and UCP respectively, for the new proposal) present better results when compared to the original ones (“iUCP_Original” and “UCP_Original”). Since there are no outliers, the maximum deviations are lower in both cases, and especially in “iUCP_Proposal” the first and second quartile are much closer to zero.

**Table 11. Absolute deviations (MRE) with the original and the new formula (per project).**

In Table 11 we compare how the new proposal affected each method’s performance. In iUCP the estimation accuracy, was better in five out of seven projects, whereas in UCP the results were worst in four out of seven.

<table>
<thead>
<tr>
<th>Project</th>
<th>Original</th>
<th>Proposal</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>44.9%</td>
<td>44.9%</td>
</tr>
<tr>
<td>P2</td>
<td>22.4%</td>
<td>3.0%</td>
</tr>
<tr>
<td>P3</td>
<td>60.6%</td>
<td>1.4%</td>
</tr>
<tr>
<td>P4</td>
<td>53.4%</td>
<td>91.8%</td>
</tr>
<tr>
<td>P5</td>
<td>128.1%</td>
<td>71.1%</td>
</tr>
<tr>
<td>P6</td>
<td>54.4%</td>
<td>8.8%</td>
</tr>
<tr>
<td>P7</td>
<td>56.5%</td>
<td>44.9%</td>
</tr>
</tbody>
</table>

=) No changes, (+) Better, (-) Worst
We further used a parametric t-test to evaluate UCP and iUCP deviation differences. According to the two-tailed significance test table [6] t(6)=2.447>-1.560, with a significance value of p=0.17 (p>0.05), is not statistically significant.

Finally using the G*Power application [11], we computed the new effect size (dz = 0.589) which is now bigger than the previously identified (dz = 0.171), configuring a large effect size. In our case this large effect of the estimation method points towards an improved accuracy achieved by iUCP over UCP. The power achieved by this new proposal is now 0.397 (which is still half of the conventional 0.8 target).

We further investigated, conducting an a-priori analysis, with an effect size of 0.589, and concluded that in order to achieve a significance of 0.05 (with a power of 0.8), the new required sample size would be 20 projects. This is yet a significant improvement when compared to the 28 projects calculated in the beginning of this section, which provide a more practical research target.

However and despite the improvements our new proposal still does not provide enough statistical evidence to support the original research hypothesis. In the next section we present our conclusions.

CONCLUSION

In this paper we presented a statistical analysis of the cost estimation results for seven real world projects using a conventional SE estimation method (UCP) and a modified HCI method (iUCP). Our research was motivated by the need to provide evidence that human-centric models can be used to enhance the software development practice, and that creating project cost estimates early in the development lifecycle is still a challenge for the software industry. Here we tried to improve the state of the art, with further empirical evidence that benefits can emerge from the cross-fertilization of the SE and HCI fields.

The main conclusion of our experiment is that we cannot statistically support the hypothesis that iUCP produces more accurate estimates than the original UCP method. This motivated a post-experiment analysis resulting in a new estimation proposal, which improves the estimation results for the iUCP method. The obtained mean deviation (34.3% for iUCP, and 69.6% for UCP) seems to provide indications that support that the interactive perspective introduced by iUCP can be more adequate for modern software development requirements. The assumption that reflecting HCI concerns into internal SE artifacts used to generate more accurate estimation models stands, and could generate value for both the SE and HCI communities.

Another relevant conclusion is that iUCP tends to estimate less effort than UCP, as denoted in the data presented both in Table 5 and Table 11. This fact is explained by the more refined assessment of actors and use cases proposed by iUCP.

Our decision to conduct a multi-organization study revealed several new issues to be tackled in the future. For instance, we found that deviations were influenced by the estimation method applicability, the lack of standardization in the use cases modeling technique, and by the real effort reporting, which is influenced both by organizational and procedural factors. On top of these factors, we also found the requirements volatility, the software development framework and the project type to play a major role on the software development team productivity, and consequently on effort estimation. This is the reason why we propose a revised formula for effort estimation.

The initial calculations using the original methods and our sample size of only seven projects resulted in a low effect size and statistical power. Therefore a sample size of 223 projects was required to achieve statistical significance. This implies a tremendous effort for future research in particular considering the difficulties obtaining data on real-world development projects. The improvements proposed here reduce the sample size to 20 projects which is an achievable target even for intensive research effort like the one required to calculate effort estimation from use-case models. This opens-up interesting perspectives for expanding this research in the future.

Future Developments

We plan to expand this experiment to a minimum of 20 projects, in order to apply both the original and the new proposal. Our goal would be to further enhance our revisions, producing a new version of iUCP that better reflects the needs of modern software development effort estimation for interactive systems.

Moreover, we plan to assess the implications of modeling interactive systems requirements using conventional and essential use cases and its implications on effort development and estimation. Furthermore, we plan to verify if there is a correlation between the project size and effort estimation deviation, in order to infer the range of project size to which the estimation methods are applicable.

ACKNOWLEDGMENTS

We thank GDAI, Logica and GMV for contributing with real projects to be analyzed. As well, author 1 gratefully acknowledges the grant from +Conhecimento (MADFDR-01-190- FEDER-000004).
REFERENCES