An Internet-Intranet Solution for Software System Estimation with Use Cases
Anca-Juliana Stoica
Royal Institute of Technology
IT University Computer and System Sciences
Forum 100 SE-164 40 Kista, Sweden
anca@is.kth.se

1. Introduction

The paper presents the results of an experimental Internet—intranet solution for assisting project management team members in software estimation by analysis based on use cases. The ESTIMATOR project was aimed at developing a complete hardware—software multi-layered Web-based system that can be used to estimate the development effort needed in a specific software-intensive project using object-oriented and UML techniques for analysis, design, and implementation [2], [4].

In a previous forum presentation [3], we identified the challenges of building complex Web Application Systems (WAS) using an approach derived from the MBASE framework [5] in order to integrate models connected to all major areas of system architecting and software engineering, namely: success, process, product, and property models. A property model for software system estimation based on use cases similar to the one presented in [6] was suggested for software system estimation early in the product life-cycle, which is applicable when OO methods, visual modeling (UML) and use case modeling [7] are used for development.

While many of the traditional concepts of software engineering hold true, WAS have a different risk profile compared to other, more mature systems. Some of the reasons are:
- shorter release cycles
- constantly changing technology and development tools
- significant security risks
- an anticipated user base that may run into a very large number from the very first day of the site’s launch.

According to the customer specification, the ESTIMATOR will be a free-service to the development community. It will collect statistical information about finished products and that information will be used to validate its internal estimation algorithm. If the estimation results differ significantly from the real results the algorithm can be tweaked to become more accurate.

The basic concepts and entities used in the algorithm correspond to the UML notation [7], in particular to the use case diagrams. This makes it easier for a user who has worked with UML for software analysis and design, to provide the information needed by the algorithm.

Use case diagrams model the dynamics of a system, along with sequence, collaboration, statechart, and activity diagrams. A use case describes a set of interactions of the things outside the current system (also called actors) with the system itself (and its key abstractions). Use cases
capture the functional requirements of a system. Actors can be humans or external systems that need some information from the current system. From an actor's perspective, a use case does something that's of value to an actor, such as calculate a result, generate a new object, change the state of an object. In order to visualize a use case and the context with other use cases, use case diagrams are used in UML. Use cases describe what the systems should do, not how they do.

Use cases allow domain experts, end users, and developers to communicate to one another. Use cases:
- capture requirements
- create value for the user
- can be used in an early stage of system development in order to assess the system's value or effectiveness
- are used in planning and controlling an iterative project
- are captured as one of the primary tasks of the Inception phase
- can be made with varying degrees of granularity, so designs should be flexible and work with whatever seems comfortable.

The ESTIMATOR project was successful in delivering in time an operational first version of an Internet-intranet hardware and software N-tier architecture for a system used to perform software estimation based on use cases. Some examples of models integrated in the MBASE approach to software system development applied to this project are:
- success: Win-Win, user-centered, decision framework (DF)
- product: UML modeling, DB modeling (object model, fully normalized into BCNF form), WAS architectural patterns, WAS artifacts (web pages, forms, framesets, targets, ..), Web enabling technologies, Web technologies (J2EE, EJB), security models
- process: Rational Unified Process (RUP)[1], major milestones (LCO, LCA, IOC), RUP artefacts, design-to-schedule
- property: use case metrics, effort and schedule estimation, scalability, usability, reliability.

In section 2 we describe the RUP-based development process for this project. Section 3 presents our version of the algorithm for software sizing based on use cases. Section 4 is a practical application of the algorithm for a real software system development, in particular our web application system. It shows how the proposed solution for software estimation was implemented, using the ESTIMATOR system for calculating the use case points (UCP) metrics. Section 5, conclusions, rounds out the paper.

II. DEVELOPMENT PROCESS

Project Information

Project Objective
Develop a multi-layered web application system for software estimation based use cases. The system was called ESTIMATOR and implemented a method for calculating use case points under the assumption that use cases are expressed in a way realizable by classes. The project was defined to be a system, including software as well as required hardware. The development approach used was MBASE/RUP.

Team Information
The ESTIMATOR project was developed at the IT University in Kista by a team consisted of 12 persons, 7 students from the software development specialization and 5 from the system development specialization. The project time span was from September until March during an
academic year. The project activities were scheduled in parallel with other educational activities. The allocated development time for the project was 3900 man-hours. In the end the project team spent more time than this on the project because a lot of the time was spent on gaining knowledge and learning new techniques.

Project Approach

The team developed the project artefacts concurrently, using a RUP process. The project was divided into phases, namely, Inception, Elaboration and Construction. The Transition phase was supposed to be implemented as a thesis work after the project was ended.

Development Process

The applied process model was RUP consisting of five iterations and three phases (Inception, Elaboration, and Construction). The five iterations were:

- **Iteration 0 (Inception):** Develop life-cycle objectives, prototype, plans, and specifications and verify the existence of a feasible architecture for the application (LCO milestone).

- **Iteration 1 (Elaboration):** Establish a specific, detailed life-cycle architecture.

- **Iteration 2 (Elaboration):** Verify architectural feasibility, and determine there are no major risks in satisfying plans and requirements (LCA milestone).

- **Iteration 3 (Construction):** Achieve a workable initial operational capability for the project.

- **Iteration 4 (Construction):** Continue development of the workable operational capability and its features, system preparation, training, use, and presentation of evolution support for the stakeholders (IOC milestone).

**A. Inception (Iteration 0)**

The specific project goals for the Inception phase were:

- define the requirements and make a basic use case model
- outline the project and define which part of the RUP process to use in order to create a good project environment.

With these goals met a good foundation was built to start from in the Elaboration phase.

The artefacts produced in this phase were:

- Software Requirements Specification
- Vision Document
- Use Case Model The Use-Case Model - described the system's requirements in terms of use cases, and was used throughout the project lifecycle. Use cases state what the system does but not how it does it. The modelling language was UML.
- Supplementary Specification
- Outcome of Use Case Seminar A use case seminar was held in order to get more information from the customer. Concepts and terms from the problem domain were gathered at this seminar and used to elaborate the use case model.

- Phase Plan
- Iteration Time Plan, Inception and first iteration in Elaboration
- Development Case - described how the project applied the RUP process. It contained the different workflows and artefacts that will be used in the project, in addition to the worker role descriptions. It also stated some general rules in the project, to make the administrative work easier.

- Risk List
- Glossary
User Interface Prototype - was a simple prototype used to give the end user a first look and feel of the application and to give feedback to the development team.

Code Template - contained the rules on how the code and comments should be written.

Project Site - was constructed in order to help out the project organisation.

An important set of activities in this project which was also a learning project for the team members, was training, basically in connection with specific courses such as: Component Technology course, to learn EJB and other technologies for distributed objects, the System Planning course, to learn about how to plan and build a hardware (network) system, the Database course, to learn database modeling and XML, and the IT Project Management, Methods, and Tools course to learn about use of modeling techniques for software and system development and management, model systems such as: MBASE and RUP, and associated tools.

In the Inception phase, WinWin negotiations with the project stakeholders about the ESTIMATOR system took place. The team worked with customers and potential users to determine their most significant needs and to reconcile these needs with a feasible implementation of architecture and plan. Mutually satisfactory (WinWin) options were adopted. A requirement team was organized, consisted of two persons that continually reviewed each other's work and cooperated in every possible way. The requirement team organized the "Use Case Seminar". The WinWin negotiation agreements were captured in the artefact "Outcome of Use Case Seminar" which was sent out to all stakeholders participants in the seminar.

The Lifecycle Objective milestone marked the end of the Inception phase was considered a success as Software Requirement Specification and Use Case Model, both approved by the customer were delivered. The team managed to create all the important artefacts as well as to gain a large amount of knowledge in many fields. Of course there were things to wish for in the end of the Inception and some of the artefacts might have been better.

B. Elaboration

The Elaboration phase consisted of two iterations. In this phase the goal was to turn the requirements into a Software Analysis. From the refined risk list and Use Case model it was decided what Use Cases were to be developed first. This enabled the team to implement high-risk functionality first and avoid possible problems later. A database analysis and a User Interface Prototype which in the end of the Elaboration should be executable were also planned.

C. Elaboration (Iteration 1)

It was defined to be a success if by its end the team had completed a System Analysis, a Software Analysis and a User Interface Analysis. Also, a basic Database Design and a Test plan were parts of this iteration. These artifacts should had to be of such quality that the second iteration of the Elaboration phase could be spent on all details and feasibility verification of a Life-Cycle Architecture and determination that there were no major risks in satisfying plans and requirements.
Key artefacts to achieve first iteration goals were:

- Software Analysis
- System Analysis
- User Interface Analysis
- Database Design
- Test Evaluation Report
- Integration Build Plan

During the Elaboration phase the documents produced by the requirements team were discussed and some important changes took place. Especially the use case models together with the Use Case Specifications were reworked. Database Design was a result of use-cases and requirements analysis. Activities performed for the design of the database were:

- Create a list over all the objects needed in the database in order to store all the information.
- Create an object model containing all the relations between the objects in the previously created object list. The model had to be fully normalised into Boyce Codd Normal Form (BCNF).
- Denormalise the object model to get a database that was optimised for this system.

System Analysis was the first step towards the design of the hardware system (network) that the project requires. The Requirement Specification, and especially the Supplementary Specification, are used to find the different needs. Software Analysis artefact was the result of the use-case analysis and a basic model of the software architecture.

User Interface Analysis artefact showed how the site was navigated as well as a brief sketch of the design of the HTML pages.

Iteration Time Plan, second iteration Elaboration was a more detailed plan, which schedules the work by dividing it into activities and artefacts (deliverables) with specific dates. There was one iteration plan for each iteration in the project, so there were two active Iteration Time Plan documents in each iteration, one for the current and one for the next iteration.

Integration Build Plan described how different software units should be integrated with the application. There was one Integration Build Plan for each build in the project.

System Architecting was done after analysing different proposals referring to what components should be part of the system. The solution is presented in Figure 1.

The architectural components were:

- Primary Domain Controller (PDC) - the central part of the system. Every network must have a PDC. This computer contains the "central" database with all accounts, passwords and access control lists.
- Firewall - protects the internal net from "attacks". That assignment can be interpreted in many different ways, such as preventing unassigned network traffic from the internet to pass to the internal net and to analyse the content of authorized traffic so that it does not contain anything that can do harm, for example viruses etc.
- Application Server - a component necessary to run the application (ESTIMATOR).
- Database Server - stores all the necessary data, both input from the users and from the application. It also contains the algorithm.
- Backup Database Server - if the Database Server "goes down" the saved information is restored from the backup files.
- Web Server - manages the communication between the user and ESTIMATOR system.
D. Elaboration (Iteration 2)

The second iteration of the Elaboration phase was considered to be a success if by its end the team had completed: System Design, Software Architecture, and User Interface Design. Also the team aimed to have a Database Design and a Test Plan along with some Test Results for the implementation of high-risk features. One of the most important results in this iteration is the first basic version of the program, Build 1. All the artefacts and the Build had to be of such quality that the first iteration of the Construction phase could be spent on implementing further Builds without wasting time on re-working architectures or designs.

System Design activity was performed in order to achieve, using the "products" provided by the stakeholder, the best performance and the best system architectural solution. The System Analysis was performed first in order to identify what the software will require from the hardware system, and then to investigate what hardware will be needed to fulfill those requirements.

Build 1 objectives were:
• to get everybody settled in with the forms planned to be used for the Construction phase
• to evaluate the process because the team wanted to test the software architecture straight through the different layers, from the user interface to the database.

Database Design was the result of the analysis of the use-cases and the requirements. It listed all the raw data that is to be stored in the database and the normalised tables.

System Analysis was the first step towards the design of the hardware system (network) that the project required. The Requirement Specification, and mainly the Supplementary Specification, were used to find the different needs. It defined what was required for the hardware system in order to have a stable platform for the software.

System Design included all the required hardware, software and also the configuration and
functionality of the network.

Software Analysis was the result of use case analysis. It was a basic model of the software architecture.

Software Architecture was the refinement of the software analysis which resulted in a detailed software architecture.

User Interface Analysis Design artefact consisted of an overall picture of how the site is navigated as well as a brief sketch of the HTML pages design.

Iteration Time Plan, first iteration Construction The iteration plan was a more detailed plan, which scheduled the work by dividing it into activities and artefacts (deliverables) with specific dates.

Integration Build Plan, build 1 described how the different software units should be integrated with the application, there was one Integration Build Plan for each build in the project.

Configuration Management Plan described the practices used when new units were added to the application as well as it defined the rules concerning version control.

Test Plan stated when and how all tests should be performed.

System Architecting basic requirements: access to four computers, two in one room and two in another room located on opposite sides of the building. The computers were hooked together in a Virtual Local Area Network (VLAN), as presented in Figure 2.

For the carrying out of the project, 192.168.19.x, was assigned, which means subnet 19. The customer requirement was to use NT-platform and that the system should be built with a budget as low as possible.

Safety was a very important issue for ESTIMATOR, in order to let the user anonymously use the system and also to enable the customer to examine the "input" and evaluate the algorithm.

The following steps were carried out in order to get a high security as part of the system design:
- Safety analysis - realistic threats and risks for the IT-system were described and, if the threats become reality, what the impact would be for the system activity.
- Safety policy - specified general safety objectives, rules and guidelines. The safety policy should also contain the division of responsibility for the safety plan.
- Safety plan - the safety policy was converted into a safety plan. Safety measures that had to be taken in order to fulfil the goals were indicated.
- Actions plan - priorities and planning of the measures that are found in the safety plan.

The system design at this iteration is illustrated in Figure 3.

In Figure 3:
- Firewall - should be between the company’s network and the Internet
- Web Server - was not part of the internal network, in order to be easily accessible from the outside (the Internet). On the other hand, a certain level of control over it was necessary, so that
no one from the outside could change the stored information. Because of this consideration, a third net was built, called demilitarised zone (DMZ). On this net, which was enabled with its own safety policy, the developers placed the Web Server as it should be easily accessible from the outside. The reason for this is to have different rules depending on whether the traffic is coming from in- or outside. Because of the way the system looked and with help from the subnet mask, subnet 19 was divided into three logical networks, one for the DMZ net, one for the firewall and one for the "internal net". Due to lack of hardware, some of the components were placed on the same computer despite the fact that this reduces safety.

- Application Server and Database Server were on the same computer in order to enhance performance. Otherwise the internal net would be heavily over-loaded as primarily these two components have to communicate with each other.

At this point the team had some suggestions about what kind of product each component should be, but nothing was definitely decided. Since the customer wanted us to use some kind of NT-platform the team decided to use Windows 2000 Advanced Server as operating system on each computer. The customer also procured Web Logic licenses, as an Application Server. So the team saw it as an obvious choice. Since cost was an important constraint the team chose Apache as Web Server since it is free and it supports all the Java components that were needed for the system. There were several options about what firewall to use, for example Firewall 1, Gauntlet or Windows 2000 Advanced Servers own packet filtering. The hardware team had to get more information before a decision could be taken. The same applied for the Database Server.

In the Elaboration phase, because the software part of the project took longer then anticipated, the hardware team started the system implementation part as well. The team installed and configured the hardware, such as network interface cards, internal memory and hard disk drives and connected the computers with network cables. The operating system installed on all servers was Windows 2000 Advanced Server. The first thing that needed to be installed was the PDC,
since that is what controls and administrates the users on the network. It contained the home
directory and the roaming profile for each user. The roaming profile made it possible to log on
to all computers and use the same desktop. This was done with Active Directory, which is a
personal database that keeps track of the settings for every user and the network. A domain
was established and the IP-configuration was carried out, accounts were set up for all users and
some useful development programs were installed. The second server to be configured was the
Application Server, mainly to see if the switch was properly configured. The Application Server
had two networks cards, one for the internal network and one with an external IP-address for the
project web site. The project site was built on JSP pages, which the institution servers did not
support, so they needed to configure their own web server. They chose to use Tomcat, since it
included the features that the project web site needed and it is a freeware program. The firewall
and the web server were connected with a twisted cable, to prevent intrusions in the data-flow.
The problems worth mentioning that were encountered when attaching the Web server to the
domain: they had neglected to enable the IP-forwarding on the firewall. On the software side, a
problem encountered in this phase was the late delivery of the UI Prototype, which impacted
the timing of designing the software architecture, based on this prototype and the use cases.

Database Design:
- a temporary database in Microsoft Access 2000 was done first
- a decision had to be taken about which DB Server to use.

UI Analysis and Design:
what platform to use; it was decided pure HTML 3.2 standard, no flash, no Java script, so that to allow a large community of users to access it
- it was decided that the Administrator UI should be different from User UI
- a minimal functionality was implemented in Explorer and Netscape (navigational map and HTML pages).

LCA Milestone
- specific software and system architectures for this application were produced
- the team elaborated the LCO artefacts further, to the level of detail required by this milestone
- an executable UI prototype was built
LCA key artefacts produced to achieve the goals at this phase were:
- Use Case Model - final
- Build 1
- Software Architecture
- DB Design
- UI Analysis and Design
- System Design

The milestone results were delayed with one week due to delayed UI Prototype.

E. Construction (Iteration 3)

In this iteration, the team started software and system implementation. The remaining use cases were analysed, designed, implemented, and tested. The main objectives were:
- refine Software Architecture to cover all use cases
- implement Build 2 and Build 3
- perform tests both on: system (hardware) and software (builds)

Design Set key artefacts produced were:
- Build 2 - a working base, to use in order to improve most functionality
- Build 3 - a simple, fully functional version of the ESTIMATOR system

Software Test Results
- System Implementation: build and configure the hardware system
- System Test Results

Database implementation was also performed, consisting of:
- searching for information about DB Servers to use - MySQL was the final choice
- configuring the server for the ESTIMATOR application
- updating the DB design.

System implementation - four computers have been used:
- Computer 1: Application Server and Database Server
- Computer 2: Primary Domain Controller (PDC) and Database Backup
- Computer 3: Firewall (WinRoute Pro)
- Computer 4: Web Server.

To make it possible for the programmers to access remotely the files located at the institution, a Virtual Private Network was installed and configured. The choice of computer for the VPN was the Application Server, because that was the only one with an external IP address. A VPN tunnel is an encrypted connection between two firewalls over the Internet. In this case it was a connection between one computer and one firewall. Figure 4 shows the System Implementation.

The software developers worked in groups of two persons, each group implementing several packages. Examples of implemented packages are: algorithm package, facade package, etc. Problems encountered: inexperience with EJB or the tools used, several glitches in the design which needed to be corrected, delays associated with these problems.
F. Construction (Iteration 4)

Activities performed were:

- fix all bugs from Construction, previous iteration
- implement the remaining use cases
- executable release of the ESTIMATOR system
- application was tested and the results included in Software Test Results artifact.

Database implementation:

- MySQL DB Server installed on Application Server configured to store all logs containing all update queries on the PDC computer (used as backup computer for the DB Server)
- script written to copy DB files to backup computer
- backup made once every 24 hours
- DB design was updated when needs for further changes were discovered.

Software implementation:

- Build 4 was the project first released operational version of the ESTIMATOR application.

The IOC package consisted of:

- code
- life-cycle documentation
- demos

The IOC Milestone marked:

- the end of Construction phase
- ESTIMATOR v1.0 was ready to be released and used.
III. SOFTWARE SYSTEM ESTIMATION WITH USE CASES

Requirements specification is a collection of documents and models to unambiguously describe the software to be built. Use cases are used to capture and express the determined system behavior and correspond to more than one scenarios, one main. Business models give input to the use-case view and help to identify actors and use cases in the system. Use cases are written in the language of the domain and are associated with sequence diagrams to show the interactions between actors and the system.

Use case analysis identifies the classes and objects for a use case's flow of events. To each use case a use case realization is associated which is a special use case that describes it in terms of the system architecture. The realization is represented by analysis-level objects and sequence diagrams.

An algorithm for estimating software system development effort based on use case metrics is presented. The algorithm is based on the Function Points (Albrecht) and Use Case Points (Karner) software metrics and an iterative risk-based object-oriented process model (like e.g.RUP). More details about applying use cases can be found in [Schneider, Winters, 1998].

Algorithm for Software System Estimation with Use Cases

1. Total actors.

\[ T_a = \sum_{i=1}^{3} (n_{a_i}) \times (W_{a_i}) \]

\( i = S / A / C. \) Actor complexity: Simple/Average/Complex.

\( n_{a_i} \) =number of actors of complexity \( i; W_{a_i} = 1 / 2 / 3; W_{a_i} = \)weight of an actor of complexity \( i. \)

Actors complexity in terms of actor interaction mechanisms: Another system with defined application interface / Communicates with the system via a protocol / Interacts through a GUI.

2. Total Use Cases.

\[ T_{uc} = \sum_{j=1}^{3} (n_{uc_j}) \times (W_{uc_j}) \]

\( j = S / A / C. \) UC complexity: Simple/Average/Complex

\( n_{uc_j} \) =number of use cases of complexity \( j; W_{uc_j} = 5 / 10 / 15; W_{uc_j} = \)weight of a use case of complexity \( j. \)

Use case's complexity in terms of transactions per U.C. or number of analysis classes / U.C.

3. Unadjusted use case points

\[ UUCP = T_a + T_{uc} \]

4. Technical Complexity of the Project
TCF = 0.6 + 0.01 \sum_{k=1}^{13} (Vtf_k) \times (Wtf_k)

13 = factors contributing to complexity (how difficult the system will be to construct).
Vtf_k = technical factor k rate related to its relevance for the project (variable): 0 = irrelevant, 5 = essential
Wtf_k = weight associated to technical complexity of factor k (fixed): Wtf_k can take values between 0.5 and 2
If all factors have Vtf_k = 3, then TCF = 1.

5. Efficiency the Project
(The Environmental Factor).

EF = 1.4 - 0.03 \sum_{l=1}^{6} (Vef_l) \times (Wej_l)

8 = factors contributing to project efficiency: familiarity with the process model used, features of the workers, difficulty of the programming language, requirements stability
Vef_l = efficiency rate for factor l, related to its relevance for the project (variable): 0 = irrelevant, 5 = essential
Wej_l = weight associated to environmental factor l (fixed): Wej_l can take values between 0.5 and 2
If all factors have Vef_l = 3, then EF = 1.

6. Use Case Points

UCP = UUCP \times TCF \times EF

7. Software System Development Effort Estimate

E = UCP \times ER [MH] [MH]=man hour.

ER = Effort Rate [MH / UCP]
IV. APPLICATION OF THE ESTIMATION ALGORITHM

Before applying the algorithm the user is supposed to have done a Use Case Model for his project. In this way he can answer the questions from the system concerning Actor complexity and Use Case complexity. Further questions to answer are about technical complexity and environmental factors.

When the user accesses the ESTIMATOR system he can choose one of two possible computation modes, namely: i) the one based on analysis classes or ii) the other one based on transactions. The difference between these two methods affects only step 2.

**Step 1 - Total Actors**

The user lists the actors in the project. For each actor he determines the type: whether it is simple, average or complex. The criteria are:

- **Simple** - the actor is another system with an interface towards the user's project system
- **Average** - the actor is another system that interacts with the user's project system through a protocol or it is a person interacting through a text-based interface.
- **Complex** - the actor is a person interacting through a GUI.

The system gives each actor type a weight as follows:

<table>
<thead>
<tr>
<th>Actor</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple</td>
<td>1</td>
</tr>
<tr>
<td>Average</td>
<td>2</td>
</tr>
<tr>
<td>Complex</td>
<td>3</td>
</tr>
</tbody>
</table>
The ESTIMATOR example

The Use Case Model of the Estimator project looks like this:
### Step 2 - Total Use Cases

The user lists the use cases of the project. For each use case he reports how many transactions or how many analysis classes it contains. Which way to choose depends on how far into the analysis phase the user has got. If the user has weighted his factors properly the outcome should be the same no matter which of the methods that is used.

A transaction is a set of activities. This set can either be performed entirely or not at all.

An analysis class is a class determined during analysis, not during design.

The ESTIMATOR system gives each use case a weight according to the schemes:

<table>
<thead>
<tr>
<th>Number of transactions</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 3</td>
<td>5</td>
</tr>
<tr>
<td>4 - 7</td>
<td>10</td>
</tr>
<tr>
<td>more than 7</td>
<td>15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number of analysis classes</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 4</td>
<td>5</td>
</tr>
<tr>
<td>5 - 10</td>
<td>10</td>
</tr>
<tr>
<td>more than 10</td>
<td>15</td>
</tr>
</tbody>
</table>

After this is done total use cases is calculated by the system.

### The ESTIMATOR example

We choose the transaction based estimation. The Use Cases can be seen in the Use Case Model in Step I. Note that only Use Cases with the highest priority are filled in this example. Furthermore, the installation of the system was solved through a third party product and is not taken into account.

<table>
<thead>
<tr>
<th>Use Case Name</th>
<th>Number of transactions</th>
<th>Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>View Estimator site</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Login</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Logout</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Maintain Project Area</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>View Glossary</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Number/weight of Classes</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Number/weight of UseCases</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>Number/weight of Transactions</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>View Project Information</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Report Result</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Get help</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Administrator Login</td>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>
Maintain algorithm 2 5

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum</td>
<td>70</td>
<td></td>
</tr>
</tbody>
</table>

\[ TUC = 12 \times 5 + 1 \times 10 = 70 \]

**Step 3 - Technical Complexity of the Project**

The user has to go through a form and rate 13 factors, each of them from 0 to 5, where 0 means that the factor is irrelevant for this project, and 5 means that the factor is essential for this project.

After this is done the ESTIMATOR system multiplies all the ratings with a weight for each factor description.

The result is then summed up and put into the following function:

\[ TCF (\text{Technical Complexity Factor}) = 0.6 \times \left(0.01 \times \text{Sum} \right) \]

**The ESTIMATOR example**

<table>
<thead>
<tr>
<th>Factor contributing to complexity</th>
<th>Rate from 0 - 5 * weight</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed system</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>Response or throughput performance objectives</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>End-user efficiency (online)</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Complex internal processing</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Code must be reusable</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Easy to install</td>
<td>3</td>
<td>0.5</td>
</tr>
<tr>
<td>Easy to use</td>
<td>5</td>
<td>0.5</td>
</tr>
<tr>
<td>Portable</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Easy to change</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Concurrent</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Includes special security features</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Provides direct access for third parties</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Special user training facilities are required</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ TCF = 0.6 + (0.01 \times 52) = 1.12 \]

**Step 4 - Efficiency of the Project**

This step calculates the environmental factor, or the experience level of the personnel in the project that is being estimated.

The user now has to go through another form and rate 8 factors, each of them from 0 to 5. What numbers 0, 3 and 5 mean is shown in the table below:

<table>
<thead>
<tr>
<th>Factor description</th>
<th>Rate from 0 - 5</th>
<th>Rating description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Familiar with Rational Unified Process</td>
<td>0/no experience</td>
<td>3/average</td>
</tr>
<tr>
<td>Application experience</td>
<td>0/no experience</td>
<td>3/average</td>
</tr>
<tr>
<td>Object-oriented experience</td>
<td>0/no experience</td>
<td>3/average</td>
</tr>
<tr>
<td>Lead analyst capability</td>
<td>0/no experience</td>
<td>3/average</td>
</tr>
<tr>
<td>Motivation</td>
<td>0/no motivation in the project</td>
<td>3/average</td>
</tr>
<tr>
<td>Stable requirements</td>
<td>0/extremely unstable</td>
<td>3/average</td>
</tr>
<tr>
<td>Part-time workers</td>
<td>0/no part-time technical staff</td>
<td>3/average</td>
</tr>
<tr>
<td>Difficult programming language</td>
<td>0/easy</td>
<td>3/average</td>
</tr>
</tbody>
</table>
After this is done the ESTIMATOR system multiplies all the ratings with the weights for each factor. The result is given by the following function.

EF (Environmental Factor) = 1.4 + (-0.03 * Sum)

---

### The ESTIMATOR example

<table>
<thead>
<tr>
<th>Factor description</th>
<th>Rate from 0 - 5</th>
<th>Weight</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Familiar with Rational Unified Process</td>
<td>1</td>
<td>1.5</td>
<td>1.5</td>
</tr>
<tr>
<td>Application experience</td>
<td>3</td>
<td>0.5</td>
<td>1.5</td>
</tr>
<tr>
<td>Object-oriented experience</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Lead analyst capability</td>
<td>2</td>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>Motivation</td>
<td>5</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Stable requirements</td>
<td>3</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>Part-time workers</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>Difficult programming language</td>
<td>4</td>
<td>-1</td>
<td>-4</td>
</tr>
</tbody>
</table>

Sum = 13

EF = 1.4 + (-0.03 * 13) = 1.01

---

### Step 5 - Use Case Points

Now the user has done his part. The ESTIMATOR system puts all the results from step 1-4 together. The result is called Use Case Points.

| Result from weighting Actors               | 8.00 |
| Result from weighting Use Cases           | 70.00|
| Result from weighting Technical Factors (TCF) | 1.12 |
| Result from weighting Environmental Factors (EF) | 1.01 |

Use Case Points = (8 + 70) * 1.12 * 1.01 = 88.23

---

### Step 6 - Project Estimate

The final estimation in the ESTIMATOR system takes the ratings in step 4 into consideration. It checks how many of the first six ratings that are below 3 and how many of the last two ratings that is above 3.

<table>
<thead>
<tr>
<th>Factor description</th>
<th>Rate from 0 - 5</th>
<th>Number of ratings below 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Familiar with Rational Unified Process</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Application experience</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Object-oriented experience</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Lead analyst capability</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Motivation</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Stable requirements</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

Sum = 2

---

19
<table>
<thead>
<tr>
<th>Factor description</th>
<th>Rate from 0 - 5</th>
<th>Number of ratings above 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part-time workers</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Difficult programming language</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td><strong>Total Sum</strong></td>
<td><strong>1</strong></td>
<td><strong>3</strong></td>
</tr>
</tbody>
</table>

The final result in this case was 3. Depending on the result different effort rates and recommendations are used as seen below.

<table>
<thead>
<tr>
<th>Number of risk areas</th>
<th>Effort rate [ MH/UCP]</th>
</tr>
</thead>
<tbody>
<tr>
<td>The final result is 0 - 2</td>
<td>Multiply the Use Case Points (step 5) with 20</td>
</tr>
<tr>
<td>The final result is 3 or 4</td>
<td>Multiply the Use Case Points (step 5) with 28</td>
</tr>
<tr>
<td>The final result is 5 or more</td>
<td>The risk of failure is high. Try very hard to make changes to your project so the numbers can be adjusted!</td>
</tr>
</tbody>
</table>

The final step of the algorithm gives the following result in man-hours [MH] for the ESTIMATOR example.

**The ESTIMATOR example**

<table>
<thead>
<tr>
<th>Use Case Points</th>
<th>Effort rate</th>
<th>= Effort [MH]</th>
</tr>
</thead>
<tbody>
<tr>
<td>88.23</td>
<td>28</td>
<td>2470.54</td>
</tr>
</tbody>
</table>

**V. CONCLUSION**

In this paper we presented experimental results obtained in building an Internet-intranet solution for software system estimation based on use cases. Two essential aspects have been emphasized:

- The development process for an operational software-hardware multi-tier Web application system
- The applicability of a use case-based effort estimation algorithm.

The experimental system was built on the theoretical foundations of the MBASE and RUP model systems which have been creatively applied in order to develop the application.

**REFERENCES**