Calculations From Other Tabs		
TCF	Technical Complexity Factor	1,02
EF	Environmental Factor	1,04
UUCP	Unadjusted Use Case Points	210
AW	Actor Weighting	15
Calculation of Use Case Points		
UCP	Use Case Points	238,7
Calculation of Estimated Effort		
Ratio	Hours of Effort per Use Case Point	24
	Hours of Effort	**5.728**
	Total price	**$42.962.400**

For additional guidance with this page, check out the following articles at Tyner Blain

Software Cost Estimation With Use Case Points - Introduction
Software Cost Estimation With Use Case Points - Final Calculations

Software Cost Estimation With Use Case Points - Free Excel Spreadsheet

Steps to Calculate Use Case Points

0 For all tabs, enter values only in the highlighted cells

1 Enter Technical Complexity Factors on the Technical tab

2 Enter Environemental Factors on the Environmental tab

3 Identify Use Cases on the Use Case tab

4 Identify Actors on the Actor tab

Technical Factor		Multiplier	Relative Magnitude (Enter 0-5)	Description
1	**Distributed System Required**	2	**0**	The architecture of the solution may be centralized or single-tenant , or it may be distributed (like an n-tier solution) or multi-tenant. Higher numbers represent a more complex architecture.
2	**Response Time Is Important**	1	**5**	The quickness of response for users is an important (and non-trivial) factor. For example, if the server load is expected to be very low, this may be a trivial factor. Higher numbers represent increasing importance of response time (a search engine would have a high number, a daily news aggregator would have a low number).
3	**End User Efficiency**	1	**5**	Is the application being developed to optimize on user efficiency, or just capability? Higher numbers represent projects that rely more heavily on the application to improve user efficiency.
4	**Complex Internal Processing Required**	1	**2**	Is there a lot of difficult algorithmic work to do and test? Complex algorithms (resource leveling, time-domain systems analysis, OLAP cubes) have higher numbers. Simple database queries would have low numbers.
5	**Reusable Code Must Be A Focus**	1	**2**	Is heavy code reuse an objective or goal? Code reuse reduces the amount of effort required to deploy a project. It also reduces the amount of time required to debug a project. A shared library function can be re-used multiple times, and fixing the code in one place can resolve multiple bugs. The higher the level of re-use, the lower the number.
6	**Installation Ease**	0,5	**5**	Is ease of installation for end users a key factor? The higher the level of competence of the users, the lower the number.
7	**Usability**	0,5	**5**	Is ease of use a primary criteria for acceptance? The greater the importance of usability, the higher the number.
8	**Cross-Platform Support**	2	**3**	Is multi-platform support required? The more platforms that have to be supported (this could be browser versions, mobile devices, etc. or Windows/OSX/Unix), the higher the value.

9	**Easy To Change**	1	**3**	Does the customer require the ability to change or customize the application in the future? The more change / customization that is required in the future, the higher the value.
10	**Highly Concurrent**	1	**4**	Will you have to address database locking and other concurrency issues? The more attention you have to spend to resolving conflicts in the data or application, the higher the value.
11	**Custom Security**	1	**3**	Can existing security solutions be leveraged, or must custom code be developed? The more custom security work you have to do (field level, page level, or role based security, for example), the higher the value.
12	**Dependence On Third-Party Code**	1	**2**	Will the application require the use of third party controls or libraries? Like re-usable code, third party code can reduce the effort required to deploy a solution. The more third party code (and the more reliable the third party code), the lower the number.
13	**User Training**	1	**5**	How much user training is required? Is the application complex, or supporting complex activities? The longer it takes users to cross the suck threshold (achieve a level of mastery of the product), the higher the value.
Calculated TCF			**1,02**	

For additional guidance with this page, check out the following articles at Tyner Blain

Software Cost Estimation With Use Case Points - Introduction

Software Cost Estimation With Use Case Points - Technical Factors

Software Cost Estimation With Use Case Points - Free Excel Spreadsheet

Environmental Factor		Multiplier	Relative Magnitude (Enter 0-5)	Description
1	**Familiarity With The Project**	1,5	**1**	How much experience does your team have working in this domain? The domain of the project will be a reflection of what the software is intended to accomplish, not the implementation language. In other words, for an insurance compensation system written in java, you care about the team’s experience in the insurance compensation space - not how much java they’ve written. Higher levels of experience get a higher number.
2	**Application Experience**	0,5	**0**	How much experience does your team have with the application. This will only be relevant when making changes to an existing application. Higher numbers represent more experience. For a new application, everyone’s experience will be 0.
3	**OO Programming Experience**	1	**4**	How much experience does your team have at OO? It can be easy to forget that many people have no object oriented programming experience if you are used to having it. A user-centric or use-case-driven project will have an inherently OO structure in the implementation. Higher numbers represent more OO experience.
4	**Lead Analyst Capability**	0,5	**1**	How knowledgeable and capable is the person responsible for the requirements? Bad requirements are the number one killer of projects - the Standish Group reports that 40% to 60% of defects come from bad requirements. Higher numbers represent increased skill and knowledge.
5	**Motivation**	1	**3**	How motivated is your team? Higher numbers represent more motivation.
6	**Stable Requirements**	2	**3**	Changes in requirements can cause increases in work. The way to avoid this is by planning for change and instituting a timing system for managing those changes. Most people don’t do this, and some rework will be unavoidable. Higher numbers represent more change (or a less effective system for managing change).

7	**Part Time Staff**	-1	**0**	Note, the multiplier for this number is negative. Higher numbers reflect team members that are part time, outside consultants, and developers who are splitting their time across projects. Context switching and other intangible factors make these team members less efficient.
8	**Difficult Programming Language**	-1	**3**	This multiplier is also negative. Harder languages represent higher numbers. We believe that difficulty is in the eye of the be-coder (groan). Java might be difficult for a fortran programmer. Think of it in terms of difficulty for your team, not abstract difficulty.
Calculated EF			**1,04**	

For additional guidance with this page, check out the following articles at Tyner Blain

Software Cost Estimation With Use Case Points - Introduction

Software Cost Estimation With Use Case Points - Environmental Factors

Software Cost Estimation With Use Case Points - Free Excel Spreadsheet

Unadjusted Use Case Points		Multiplier	Number of Use Cases	Description
1	**Simple**	5	**40**	Simple Use Case - up to 3 transactions.
2	**Average**	10	**1**	Average Use Case - 4 to 7 transactions.
3	**Complex**	15	**0**	Complex Use Case - more than 7 transactions.
Calculated UUCP			**210**	

Individual Use Cases		Multiplier	Use Case Name
1	Simple	5	Agregar Estación
2	Simple	5	Eliminar Estación
3	Simple	5	Modificar Estación
4	Simple	5	Agregar Bus
5	Simple	5	Eliminar Bus
6	Simple	5	Modificar Bus
7	Simple	5	Agregar Medida
8	Simple	5	Cambiar Estado Solicitud
9	Simple	5	Tomar Medida
10	Simple	5	Consultar Solicitudes por ticket
11	Simple	5	Consultar todas las solicitudes
12	Simple	5	Realizar Solicitud
13	Simple	5	Agregar Ruta
14	Simple	5	Editar Ruta
15	Simple	5	Eliminar Ruta
16	Simple	5	Consultar Todas Rutas Existentes

17	Simple	5	Consultar Rutas que pasan por Estación
18	Simple	5	Adquirir Tarjeta
19	Simple	5	Consultar Saldo de Tarjeta
20	Simple	5	Modificar Datos Personales
21	Simple	5	Ingresar al Bus
22	Simple	5	Recargar Tarjeta
23	Simple	5	Agregar Tarjeta
24	Simple	5	Modificar Tarjeta
25	Simple	5	Eliminar Tarjetas
26	Simple	5	Autenticar
27	Simple	5	Agregar Director
28	Simple	5	Agregar Pasajero
29	Simple	5	Agregar Operador
30	Simple	5	Agregar Conductor
31	Simple	5	Agregar Auxiliar
32	Simple	5	Eliminar Director
33	Simple	5	Eliminar Operador
34	Simple	5	Eliminar Auxiliar
35	Simple	5	Eliminar Conductor
36	Simple	5	Modificar Director
37	Simple	5	Modificar Auxiliar
38	Simple	5	Modificar Conductor
39	Simple	5	Modificar Operador
40	Simple	5	Modificar Datos
41	Average	10	Generar Reporte
Insert additional rows above this row and copy the cell values to automatically update the counts of actors by type			

For additional guidance with this page, check out the following articles at Tyner Blain

Software Cost Estimation With Use Case Points - Introduction

Software Cost Estimation With Use Case Points - Use Case Analysis

How to Write Good Use Case Names - 7 Tips

Software Cost Estimation With Use Case Points - Free Excel Spreadsheet

Actor Summary		Multiplier	Number of Actors	Description
1	**Simple**	1	**0**	Simple actors are other systems that communicate with your software via a pre-defined API. An API could be exposed through a dll, or as a REST, SOAP, or any web-service API or remote procedure call (RPC). The key element is that you are exposing interaction with your software through a specific, well-defined mechanism.
2	**Average**	2	**0**	Average actors can either be human beings interacting in a well defined protocol, or they could be systems that interact through a more complex or flexible API.
3	**Complex**	3	**5**	The original definition of complex actors specifies that users who interact with the software through a graphical user interface are complex actors. While that is true, the same classifcation should apply to users who interact with the system in unpredictable ways. An AJAX interface that exposes more of the underlying application (and data stores) than would be available through a rigid protocol might introduce similar complexity.
Calculated AW			**15**	

Individual Actors		Multiplier	Actor Name
1	Complex	3	Director
2	Complex	3	Administrador
3	Complex	3	Pasajero
4	Complex	3	Auxiliar
5	Complex	3	Operador
Insert additional rows above this row and copy the cell values to automatically update the counts of actors by type			

For additional guidance with this page, check out the following articles at Tyner Blain

Software Cost Estimation With Use Case Points - Introduction

Software Cost Estimation With Use Case Points - Actor Analysis

Software Cost Estimation With Use Case Points - Free Excel Spreadsheet