CAST Enhancement Function Points Estimation - EFP

On this page:

Target audience:

CAST AI Administrators

Introduction

CAST Enhancement Function Points Estimation technology is an automatic function points counting method based on the rules defined by the OMG. This automatic process is described in detail in CAST OMG-compliant Automated Function Points. After an initial baseline count, CAST AIP catalogs and stores both Data Functions and Transactional Functions structure in the CAST Application Intelligence Knowledge base. All artifacts that are part of the Data and Transactional Functions are stored into the database along with a checksum that enables uniquely identify the current version of each artifact.

When a new version of the application source code is analyzed using CAST AIP, the CAST Enhancement Function Points algorithm uses the computed checksum of each artifact composing both data functions and transactional functions to determine:

- Added Functions
- Modified Functions
- Deleted Functions

Then each Added, Modified or Deleted data functions and transactional functions are weighted using OMG tables.

Note that for new Applications on-boarded in this release of CAST AIP, the AEP mode is used by default. The EFP measurement mode can be used instead by specifically selecting it in the CAST Management Studio > Application editor > Function Points tab. Note that changing the mode for an Application that has snapshots calculated with the default AEP mode will impact snapshot results.

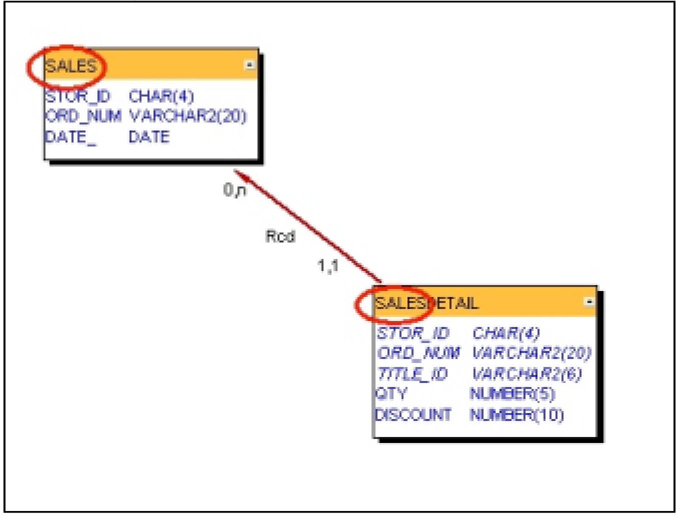
Data & Transactional Functions

Data Functions

From the baseline Function Points, CAST AIP catalogs and stores Data Functions and their structure. Via the default algorithm, Data Functions are usually based on SQL Tables. The algorithm groups SQL tables into Data Functions using both naming convention and primary key - foreign key relationships. Whenever the SQL tables do not use referential integrity based on primary key - foreign key relationships or do not use any naming convention, then each table will be counted as one data function by the algorithm.

However, when the application uses both primary key - foreign key relationships and naming convention, the algorithm grouping table in a way that is very similar to the identification of Master-Details tables.

Each Data Function is uniquely identified by CAST thru its Master table and the Master table will be used to determine Added and Deleted Functions. In the CAST Enlighten screenshot below, the master table is the Sales SQL table:



SALES

```
STOR_ID   CHAR(4)
ORD_NUM   VARCHAR2(20)
DATE_     DATE
```

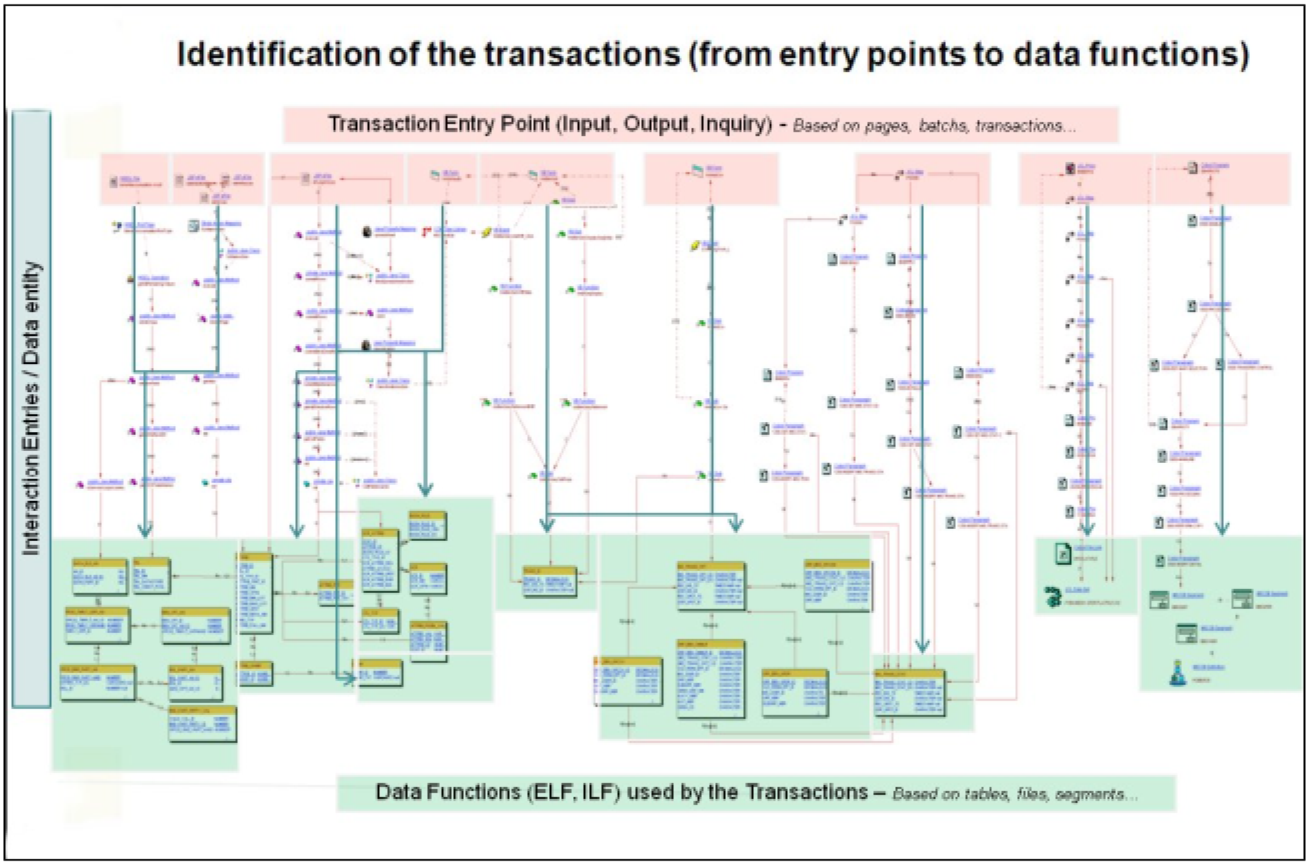
SALESDETAIL

```
STOR_ID   CHAR(4)
ORD_NUM   VARCHAR2(20)
TITLE_ID  VARCHAR2(6)
QTY       NUMBER(5)
DISCOUNT  NUMBER(10)
```

Transactional Functions

From the baseline Function Points, CAST AIP catalogs and stores Transactional Functions and their structure. Via the default algorithm, Transactional Functions are identified using an impact analysis algorithm that searches, starting from the transaction entry point and using the artifact dependencies, all the data functions used directly or indirectly by the transaction entry point.

CAST identifies the transaction entry points as well as all the technical objects used to implement the data processing:



Identification of the transactions (from entry points to data functions)

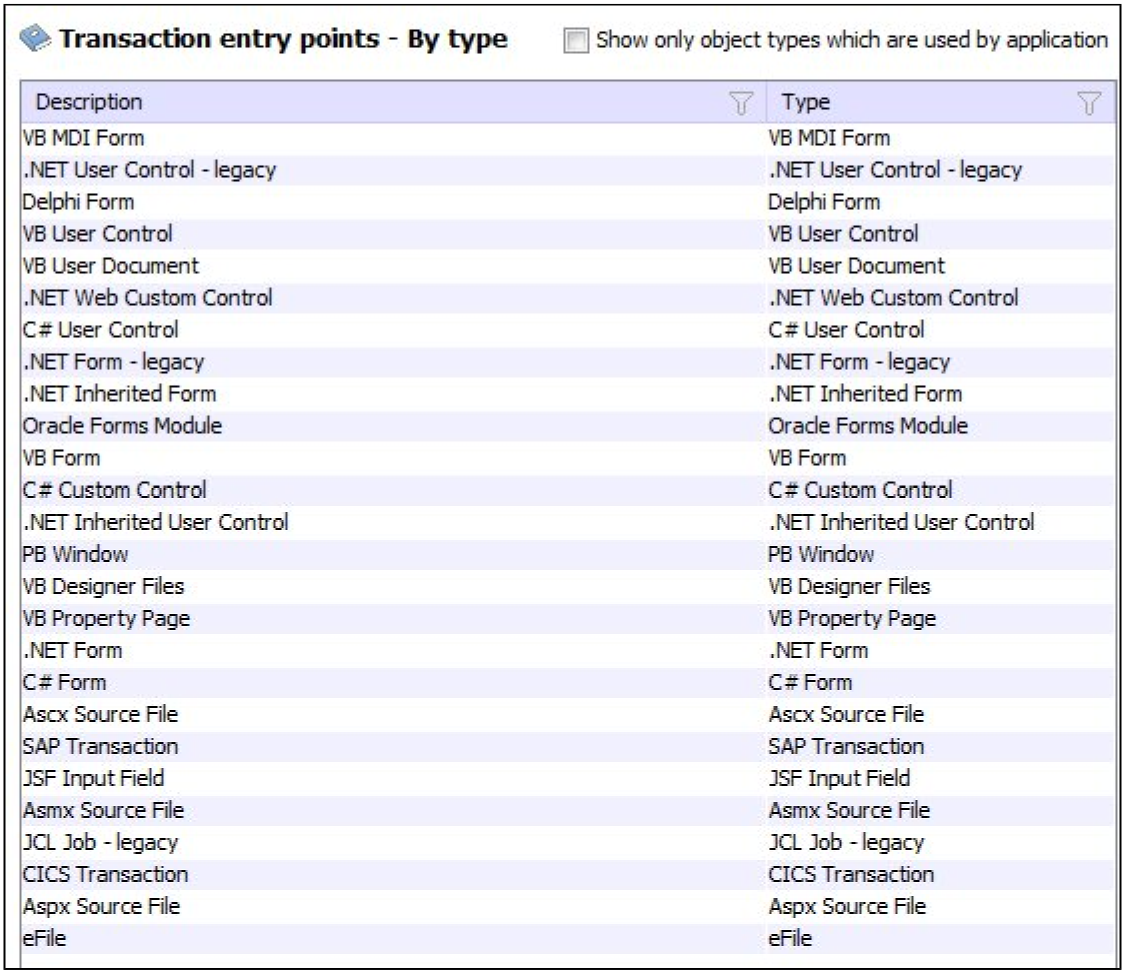
Transaction Entry Point (Input, Output, Inquiry) - *Based on pages, batchs, transactions...*

Interaction Entries / Data entity

Data Functions (ELF, ILF) used by the Transactions – *Based on tables, files, segments...*

Each Transactional Function or transaction is uniquely identified by CAST AIP through its transaction entry point. Transaction entry points are usually the default user forms as defined in each programming language or can be chosen by CAST AIP's user within the **CAST Transaction Configuration Center,** as explained in detail in CAST OMG-compliant Automated Function Points.

Here is the list of the default transaction entry points types:

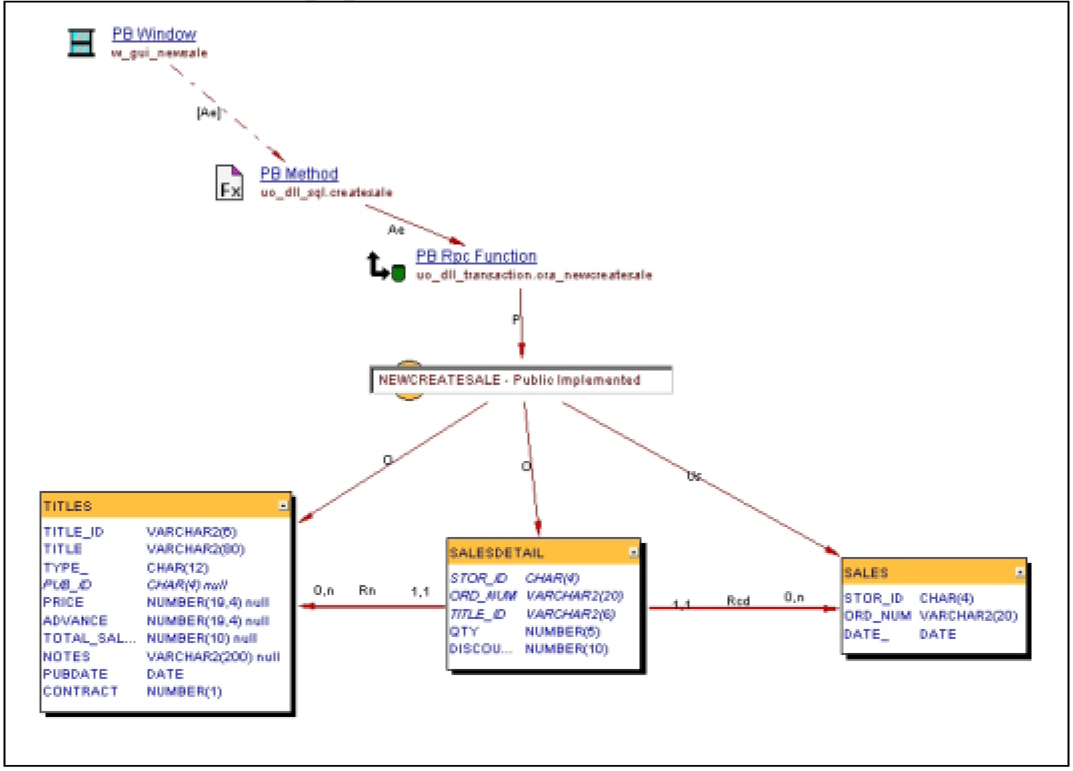


Transaction entry points - By type

□ Show only object types which are used by application

Description	Type
VB MDI Form	VB MDI Form
.NET User Control - legacy	.NET User Control - legacy
Delphi Form	Delphi Form
VB User Control	VB User Control
VB User Document	VB User Document
.NET Web Custom Control	.NET Web Custom Control
C# User Control	C# User Control
.NET Form - legacy	.NET Form - legacy
.NET Inherited Form	.NET Inherited Form
Oracle Forms Module	Oracle Forms Module
VB Form	VB Form
C# Custom Control	C# Custom Control
.NET Inherited User Control	.NET Inherited User Control
PB Window	PB Window
VB Designer Files	VB Designer Files
VB Property Page	VB Property Page
.NET Form	.NET Form
C# Form	C# Form
Ascx Source File	Ascx Source File
SAP Transaction	SAP Transaction
JSF Input Field	JSF Input Field
Asmx Source File	Asmx Source File
JCL Job - legacy	JCL Job - legacy
CICS Transaction	CICS Transaction
Aspx Source File	Aspx Source File
eFile	eFile

The transaction entry point will be used to determine Added and Deleted Functions. In the CAST Enlighten screenshot below, the transaction entry point is the PowerBuilder window called w_gui_newsale:



TITLES	
TITLE_ID	VARCHAR2(6)
TITLE	VARCHAR2(80)
TYPE_	CHAR(12)
PUB_ID	CHAR(4) null
PRICE	NUMBER(19,4) null
ADVANCE	NUMBER(19,4) null
TOTAL_SAL...	NUMBER(10) null
NOTES	VARCHAR2(200) null
PUBDATE	DATE
CONTRACT	NUMBER(1)

SALESDETAIL	
STOR_ID	CHAR(4)
ORD_NUM	VARCHAR2(20)
TITLE_ID	VARCHAR2(6)
QTY	NUMBER(5)
DISCOU...	NUMBER(10)

SALES	
STOR_ID	CHAR(4)
ORD_NUM	VARCHAR2(20)
DATE_	DATE

Added Data & Transactional Functions

Based on the identification of the differences between 2 application versions, the CAST Enhancement Function Points algorithm identifies, the data entities that have a new master table as added data functions. Similarly, the Enhancement Function Points algorithm identifies the transactions that have a new transaction entry point as added transactional functions. The master table and transaction entry point are considered as new whenever these artifacts did not exist in the previous application version with the same name, type and full qualified name.

Deleted Data & Transactional Functions

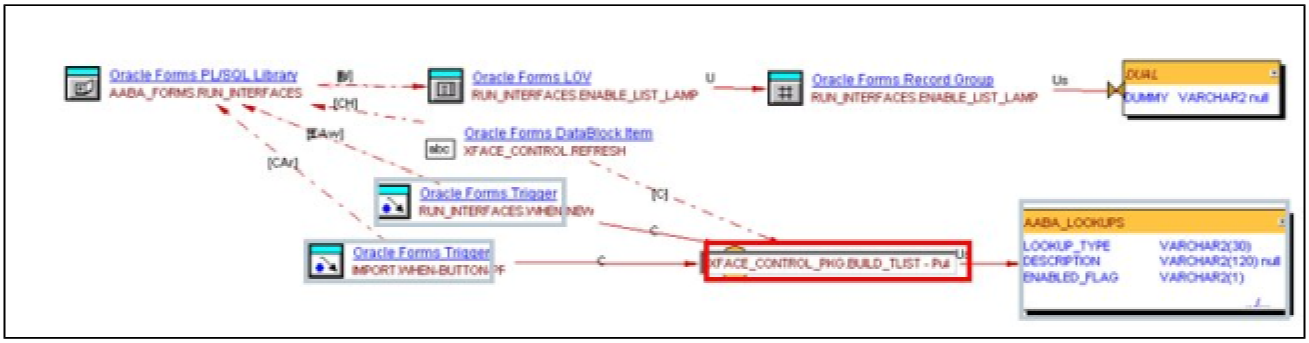
Based on the identification of the differences between 2 application versions, the CAST Enhancement Function Points algorithm identifies, as deleted data functions, the data entity's master table that no longer exist in the new application version analyzed. Similarly, the Enhancement Function Points algorithm identifies, the transactions those entry point no longer exist in the new application version analyzed as deleted transactional functions. The master table and transaction entry point are considered as no longer existing whenever these artifacts do not exist in the new application version with the same name, type and full qualified name.

Modified Data & Transactional Functions

CAST Enhancement Function Points algorithm identification of the modified data and transactional function is based on the structure, the objects and dependencies, of each data entity and transaction. A Data Function is considered as modified if at least one of the SQL tables has been modified. An SQL table is modified if columns have been added or modified by a change of the name of the column or its type. The changes of the indexes are not considered as functional changes and, as such, a table that has an index change is not considered as modified. Transactional Function is considered as modified if its entry point; the data functions it uses and/or any of the code elements used or called along the transaction are changed. As part of the transaction, CAST integrates all components the transaction uses including technical shared components. One should be especially cautious with shared technical components, when such shared components are modified, many transaction that rely on them will be identified by the CAST algorithm as changed.

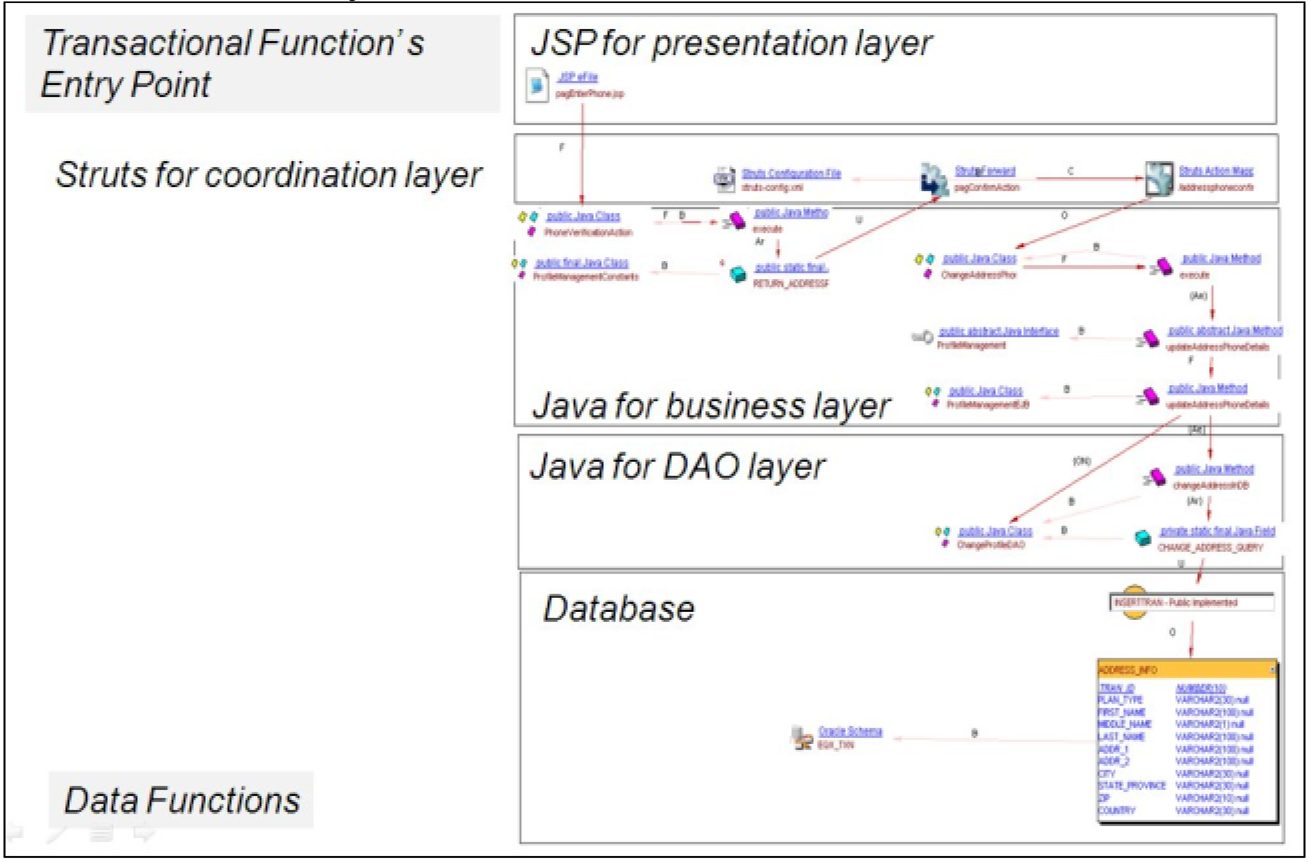
Entry points and code elements are detected by the source code analyzers as changed if their signature is changed and/or their source code is changed. A change of the comments or blank lines is not taken into account.

Here is an example of a transactional function from an Oracle application. In this case, the stored procedure XFACE_CONTROL_PKG.BUILD_TLIST that embeds several SQL queries is detected as modified, then the transaction it belongs to will be considered as modified:


Oracle Forms PL/SQL Library
AABA_FORMS.RUN_INTERFACES
[M]
[CH]
[EAw]
[CAr]
Oracle Forms LOV
RUN_INTERFACES.ENABLE_LIST_LAMP
U
Oracle Forms Record Group
RUN_INTERFACES.ENABLE_LIST_LAMP
Us
DUAL
DUMMY VARCHAR2 null
Oracle Forms DataBlock Item
XFACE_CONTROL.REFRESH
[C]
Oracle Forms Trigger
RUN_INTERFACES.WHEN-NEW
C
Oracle Forms Trigger
IMPORT.WHEN-BUTTON-PF
C
XFACE_CONTROL_PKG.BUILD_TLIST - Pul
Us
AABA_LOOKUPS
LOOKUP_TYPE VARCHAR2(30)
DESCRIPTION VARCHAR2(120) null
ENABLED_FLAG VARCHAR2(1)

As a consequence, if the stored procedure XFACE_CONTROL_PKG.BUILD_TLIST is used in several transactional functions then these transactional functions will also be considered as modified, following the change in behavior of the stored procedure.

Another example of a transactional function from a J2EE & Oracle application with several layers written in different languages: An enhancement in such environment might be implemented in any of the layers: Presentation, Coordination, Business and Data Access as well as the database itself. The Entry Point of the transaction is the JSP page called pageEnterPhone.jsp while the coordination layer is implemented using several Struts artifacts and the business layer is implemented thru several Java classes. Changes of one of these JSP files, methods, or stored procedures will result in CAST Enhancement Function Points classifing the transaction as modified.



Weighting and Final Calculation

CAST Enhancement Function Points generates Unadjusted Function Points. The calculation of the weight of the data functions and transactional function is based on OMG-compliant Automated Function Points tables. The official OMG-compliant Automated Function Points formula for the Enhancement Function Points is as follows:

```
EFP = [(ADD + CHGA + CFP) * VAFA] + (DEL* VAFB)
```

Where:

- **EFP** is the enhancement project function point count.
- **ADD** is the unadjusted function point count of those functions that were added by the enhancement project.
- **CHGA** is the unadjusted function point count of those functions that were modified by the enhancement project.
- **CFP** is the function point count added by the conversion.
- **VAFA** is the value adjustment factor of the application after the enhancement project.
- **DEL** is the unadjusted function point count of those functions that were deleted by the enhancement project. It is important to consider the absolute value of the deleted FP cannot have a negative value.
- **VAFB** is the value adjustment factor of the application before the enhancement project.

CAST generates the **ADD**, **CHGA** and **DEL** numbers using the OMG-compliant Automated Function Points tables below.

The weight and the associated number of function points of each internal data function (ILF) is given by the following table (DET stands for Data Element Type and RET stands for Record Element Type):

	1 to 19 DET	**20 to 50 DET**	**51 or more DET**
0 or 1 RET	Low = 7 FP	Low = 7 FP	Average = 10 FP
2 - 5 RET	Low = 7 FP	Average = 10 FP	High = 15 FP
6 or more RET	Average = 10 FP	High = 15 FP	High = 15 FP

A similar table is available for EIF. The number of function points differs slightly for Internal (ILF) and External (EIF) data functions:

	Internal (ILF)	External (EIF)
Low	7 FP	5 FP
Average	10 FP	7 FP
High	15 FP	10 FP

For transactional functions, CAST uses also 2 OMG-compliant Automated Function Points tables as computer-based algorithm is cannot judge the primary intent of a GUI or a transaction. The weight and the associated number of function points of each transactional function is then given by the following table (this one is for EI):

	1 to 4 DET	5 to 15 DET	16 or more DET
0 or 1 FTR	Low = 3 FP	Low = 3 FP	Average = 4 FP
2 FTR	Low = 3 FP	Average = 4 FP	High = 6 FP
2 or more FTR	Average = 4 FP	High = 6 FP	High = 6 FP

Similar tables are available for EO and EQ. The number of function points differs slightly for each type of transactional functions:

	EI	EO	EQ
Low	3 FP	4 FP	3 FP
Average	4 FP	5 FP	4 FP
High	6 FP	7 FP	6 FP

CAST AIP with conservatively uses the table External Output (EO).