Process Overview

Preamble to the AFP on-boarding process

Why an application on-boarding/on-going process?

Business critical applications range from complex to very complex, in terms of architecture, technologies, structural components, and integration with other applications or operating environment. The analysis of such applications should take in to consideration all application dimensions and it is anything but *plug and play.* This is particularly true for applications that include custom "home-grown" technologies like, for instance, non-standard scripting languages.

Application design and development takes time and requires a complex process to manage often large and distributed teams. It includes multiple project milestones and phases, unit testing, QA and RAID (Risks, Assumptions, Issues, Dependencies) assessments at each step of the SLDC process. Moreover, projects based on Agile methodology require to deal with interdependent stories, and a strict process is needed to manage the development sprints and the integration of parallel development branches. A correspondingly rigorous process is necessary to analyze applications in order to effectively deal with this complexity.

A well-defined process begets a well-defined analysis. The more details and preciseness built into the process, the better the ultimate end result and, of course, the accuracy and value delivered by the AIP dashboard. The better and more exact the analysis represents the state of the application, the easier it will be for application development and management teams to understand what they have to adjust.

What are the expected benefits?

The first benefit is to get a CAST AIP dashboard that shows accurate analysis results and metrics. This includes accurate and consistent variations in key metrics (business criteria, health factors application and enhanced Function Points) between releases. In addition to that, a formal process to on-board applications and configure analysis provides the assurance that the analysis scope truly reflects and matches the whole application.

The second benefit is also the complete identification of all applicative transactions through the accurate review of the technical architecture, the handling of frameworks, and the list of all non-standard technologies. Identifying applicative transactions is an important step to counting Function Points accurately.

Lastly, a formal on-boarding process allows more realistic planning and supports more accurate analysis completion estimates tied directly to the application complexity with regards to the technologies used for implementation.

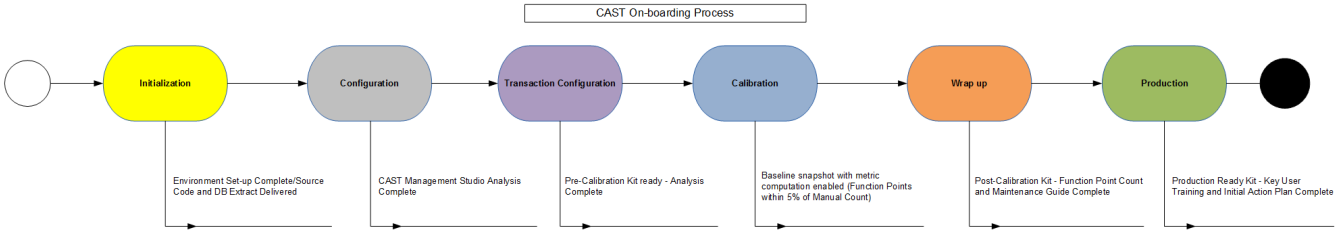
What are the pitfalls to avoid?

A formal and strict on-boarding process helps avoid common pitfalls that may lead to incorrect or partial results:

- Lack of knowledge and experience: Analysis settings can be defined incorrectly and technologies used in a specific part of applications can be missed, impacting the results. This issue is generally fixed by training the AI administrators regularly.
- Insufficient source code delivery procedures: Those are not based on information collected in the "Application Technical Survey" (ATS) and the "Source Code Delivery Form" (SCDF), and can impact the application boundaries and associated analysis scopes.
- Inaccurate analysis effort estimation, planning, and project milestones: The credibility of the Analysis Intelligence Center and the information it delivers is decreased.

The main steps

The AFP process is implemented in CAST AIP through several steps, as presented in the following figure:



Initialization

This first step aims to organize communication between actors and to define roles and responsibilities. The kick-off meeting allows to present and describe the sequence of events and the expected output to be produced.

At the end of this step, source code extraction prerequisites (ATS, SCDF, and source code extractors) should have been defined and validated by the Application team and the application must have been registered in the AIC Portal.

Configuration

The objective of the second step is set-up CAST AIP, to specify the different modules that will be populated, and to define the rules to check the source code delivery with regards to the application boundary.

This step is generally done once but can be re-executed in case of changes in source code delivery or application analysis settings. Modification of the application boundary should be described in a change order request.

Transaction configuration

The third step is focused on the rules to identify transactions: Transaction Entry Points, Data Entities, and Transaction End Points. Note that all actions to create missing links due to a specific technique of development will be considered as a configuration task. Using predefined rules for Transaction Entry Points and Transaction end Points saved in a calibration kit can help the AI administrator in charge of the transaction configuration.

The output of the transaction configuration step is the list of complete transactions and a justification for all incomplete transactions (an incomplete transaction does not have any value in the "computed" columns of TCC Result/Transactions view). The list of transactions and corresponding persistent objects should be shared with the Application team in order to get an acknowledgement of the number of interfaces for the application.

Calibration

This step is focused on the grouping, the splitting, and the exclusion of transactions as well as the overwrite of the calculated value. At that point of the process, all transactions should have been identified and validated by the Application team. Calibration kits will help to apply rules for the different actions of calibration operation.

Wrap-up

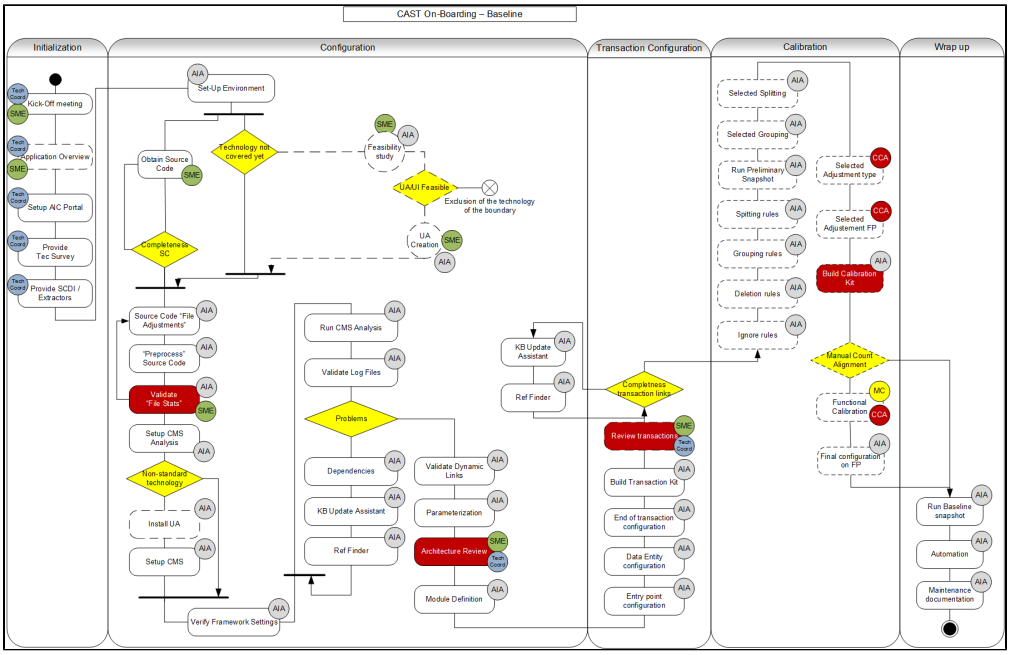
The objectives of this step are to produce the list of applications to be registered in the on-boarding process, to automate the process, and to describe the required actions for the on-going measurement.

Production

This last step is about all the operations described in the maintenance kit to perform a new analysis of the application.

The on-boarding process in details

This part provides more details regarding the different steps of the AFP on-boarding process. The following figure presents the whole process.



Initialization

The Kick-Off meeting

The goal of this meeting is to present to all the involved actors the objective of the measurement program, the process, the time-line of the activities on the application, the degree of involvement of the SME or people in charge of the application. As a takeaway of this meeting, role and actors should be clearly identified and the expected workload for the Application team shared with all actors.

The Application On-Boarding Kick-Off Meeting brings together for the first time all the players involved in the configuration and the analysis of an application by the AIP platform. The following people should be present at this meeting:

- AIP Project Manager
- AI Administrator
- Application Development Manager(s)
- Application team / Subject Matter Expert(s)

Application Overview meeting

During the Application Overview meeting, the AI Administrator meets the Application team and the Subject Matter Expert (SME) to validate the technical survey and define how source code and database extraction will be delivered. Moreover, issues related to source code delivery (source files, location, access rights, ...) should be resolved during this meeting.

The meeting focuses on the application for which the Application team can share relevant characteristics such as the technologies in place, specific frameworks used, or any other information that can help to decide to stop or continue.

Important points to take in consideration that can impact the planning:

- SME should participate in the meeting to discuss the technologies used in the application. In case of various technologies (e.g. mainframe Cobol /DB2 and Java/Oracle), it is often better to organize several meetings (ideally one for each technology).
- If the application accessed a DBMS like MS SQL Server or IBM DB2, keep in mind that setting up an environment that is similar, in terms of database structures, to the one in production is time consuming.

Register the application

Once the application and the associated delivery process have been validated with the SME and the Application team, the AI technical coordinator registers the application in CAST AIC Portal and configures the DMT tool to collect and deliver the source code to the CAST AIP platform.

For more information about registering domains and applications in AIC Portal, refer to the CAST AIP documentation: 1.1. Registering new domains and applications in the CAST AIC Portal.

Provide the Application Technical Survey

The Application Technical Survey helps qualifying applications and addresses multiple systems, types of applications, and technology usage questions. This document aims to gather information related to the implementation of the application (technologies and frameworks used, specific techniques of development, ...). It will help select the candidate application and identify new technology silos (a new silo requires the preparation of a new specific configuration).

The Application Technical Survey should be presented to the Application team and the SME during the Kick-Off meeting and the Application Overview meeting is often a good opportunity to fill it in.

For more information about application qualification, refer to the CAST AIP documentation : 1.3. Application Qualification.

Provide Source Code Delivery Instructions and Extractors

The Technical Coordinator provides the Application team with source code extractor (downloaded from the AIC Portal) - the CAST Delivery Manager Tool, and instructions about the source code delivery process. He can also support the SME to configure the DMT tools.

For more information about configuring the DMT, refer to the CAST AIP documentation and the CAST University courses.

Configuration

Prepare the environment

Create the directory structure that is going to host the source code and the database extractions. You can use the following folders:

- source_provided
- source_delivered
- analyzed

A Sandbox environment could be particularly useful to fine tune the CAST AIP configuration and define the future production environment (for the on-going mode).

Technology review

At this step, it is important to identify all the technologies used in the implementation of the application. In case a technology is not supported (i.e: there is no AIP analyzer and no Universal Analyzer configuration) and is used to implement a large part of the components, a solution is to postpone the on-boarding of the application. Indeed, **a custom support of a technology should not be part of the critical path of the project**.

If a technology is not supported and there is no available Language Extension, then the support through a custom development can be studied with CAST consultants and SME.

The conclusion of that study must help decide if the custom development can be done or not. If this is too expensive or if the impact of the technology on the application boundary is too low compared to the development cost, then the support should not be done. If the application is based on several technologies and some of them are not supported, then a good solution is to postpone the on-boarding to avoid impacting the critical path of the project. Note: There are multiple techniques to draw links across the components of a given technology. These should taken in consideration in the study.

If the study provides a positive conclusion, then using Universal Analyzer can be considered to provide more transactions to Function Point counting.

Deliver the source code

Once the application has been registered and the list of technologies has been validated, the source code can be delivered by the Application team to the AI administrator.

For more information on how to deliver the source code, refer to the CAST AIP documentation: 1.4. Deliver the application source code version

> The source code that has been delivered will represent the application boundary. As a consequence, pay a particular attention to components that could have been missed.

Once the application boundary has been delimited, additional pieces of code should be used as improvements of the application analysis configuration.

Source code clean-up

In order to avoid polluting the application analysis results with non relevant information, the source code in Source_Delivered sub-folders should be cleaned-up. Remove source files dedicated to test .

Source code preprocessing

In some situations, the source code cannot be parsed as such and must be preprocessed. This operation is done by a dedicated tool in either Source_Delivered or Analyzed sub-folders. A study should be done first to identify the preprocessing rules that must be applied to the source code .

Check source files

It is often interesting to list all the file extensions that are present in the source code folders. Some of them are not used during the application analysis and can denote a particular technology that has been missed at the qualification time. For example, "cxx" files correspond to C++ technology.

Note that if the number of files that have not been taken in account is low, then the corresponding technology can be ignored. Nevertheless, this point should be discussed with SME .

Deploy Language Extensions for non-supported technologies

At this step, the different Language Extension configurations required to support non supported technologies and/or frameworks that are part of the application must be deployed. This should not consider the development of new Language Extensions that can impact the application analysis planning.

Configure analysis settings in CAST Management Studio

The Analysis Unit must be set up for the technology they correspond to. For instance, for a JEE Analysis Unit, it is important to verify if the version of the JDK and the frameworks (such as Hibernate, Spring, ...) have been configured properly.

Universal Analyzer, Universal Importer configurations, and specific preprocessing routines that have been identified during the Architecture Review meeting can be added at application-level.

For more information on how to define analysis setting, refer to the CAST AIP documentation: 2.1. Set-up the analysis

Run the analysis and check logs

Once the analysis settings have been configured, analyze the application and review the different logs that have been produced. This is an important step since it allows to identify missing pieces of code (see messages like "could not resolve 'xxx' in import 'com.xxx.yyy'") or missing framework configuration files (ex: struts-config.xml) that will impact the number of transactions. Resolving these issues is generally done by adjusting the analysis settings and running analysis again (for the whole application or for some Analysis Units only).

For more information on analysis and log review, refer to the CAST AIP documentation:

- 2.2.1. Run the Analysis
- 2.2.2. Validate and fine tune the Analysis

Dealing with missing and unexpected links

Dependencies check

AI Administrator should review all the dependencies between the Analysis Units of the application in order to ensure the configuration is ready for the next stage.

KB Update Assistant

If some adjustments are required after the first analysis, the KB Update Assistant tool can be used. It provides a framework to add or ignore some links based on dedicated queries that can be implemented by the AI Administrator. However, check the customization library before creating new queries.

Reference Pattern

Reference Pattern can also be an option. However, you must pay attention to the false links you may generate with this approach.

Validate Dynamic Links

Once analysis is complete, validating dynamic links plays a key role in preparing the final configuration of transactions and the generation of the snapshot. The analysis engine will draw dynamic links between technical objects. Even if most of them will be valid, it is often necessary to review the list and to exclude unexpected dynamic links by using the Dynamic Link Manager.

Parameterization

The validation of dynamic links can require the adjustment of the parametrization settings. This allows to define new exclusion patterns that will be applied at execution time.

This very powerful feature should be considered if you have a large number of dynamic links. Most of the time a high number of dynamic links means a specific framework has not been taken in account. In this case, you should identify some dynamic links which have been created from some strings passed to methods of a logger component. You can then ignore the activation of the inference engine.

Architecture Review

If this is the first on-boarding of the application, an Architecture Review meeting should be held with the Application Team and the Subject Matter Expert (s). The purpose of this meeting is to confirm the list of the technologies for the application and to identify the interfaces between them. Critical points must be addressed during this meeting, such as:

- Identification of key source code files that represent the interfaces between the application and the database. The SMEs should identify these files by their name.
- Mapping files for missing frameworks (ex: Ibatis). The SMEs should identify these files by their name.
- Identification of Application entry points (ex: shell scripts that load data into the database via tools like Oracle's SqlLdr or SqlPlus, a hosted web service which returns data to request, a daemon listener class which processes various status requests, ...). SMEs should identify corresponding files that play a role in these transactions.

Interaction between technologies:

- **Technology and framework interaction:** List all technologies involved in the application with their interactions per layer and per module
- **Interfaces review:** List all technologies with their interactions with end-users or third-party applications.
- **Data Storage review:** List all technologies with their interactions to data storage.

Interfaces review :

- **Technology and framework review:** List all technologies involved in the application
- **Interfaces review:** List all technologies used to interact with end-users or third-party applications.
- **Data Storage review:** List all technologies used to store data.

> If specific code elements are used as User Interface or to expose services, then the list must be determined and validated. For instance, Web Services can be used to expose services to internal modules of the application or to other applications. In this case, only these code elements will be considered as transaction entry points.

Module definition

Modules are used to define which code elements should be part of the application and which should not. For analysis reasons it can be necessary to analyze more source code than is in the initial scope of the application and, in this case, the module definition will allow to define the subset of the components that will define the application boundary.

Transaction configuration

Transaction Entry Point configuration

Use the CAST Transaction Configuration Center to adjust transaction settings according to the information provided by the Architecture Review meeting. The input of this step is the section of Architecture Review Document that describes the application interfaces with external entities. An entity can be either a person or a another application. As a consequence, consider the User Interfaces an application can have as well as the API or Web Services the application exposes.

For more information on how to configure transactions, refer to the section: Transaction Configuration.

Data Entity configuration

Use the CAST Transaction Configuration Center to adjust Data Entity settings according to the information provided by the Architecture Review meeting. The input of this step is the section of the Architecture Review Document that describes the application interfaces with persistence layers. These ones must be considered to complete the transaction configuration.

For more information on how to configure Data Entities, refer to the section: Transaction Configuration.

Transaction End Point configuration

Once Transaction Entry Points and Data Entities have been configured, use the CAST Transaction Configuration Center to identify Transaction End Points and tune the configuration.

Empty transactions should be validated. The transaction builder engine considers a transaction as valid if it starts from an Entry Point and reaches Data Entities and/or predefined End Points. You can define additional rules to select Transaction End Points, based on the interfaces, API, and libraries that have been identified.

You can also refine transaction configuration further by reusing and merging TCC Configuration.

Build the transaction kit and review transactions

When the transaction configuration is complete, you can export the list of transactions and Data Entities that have been selected, and share them with the SME. It is important to validate the content of this list and to check that no element is missing.

For more information on this part, refer to the section: Transaction Review.

If transaction scopes must be improved, then go back to the CAST Transaction Configuration Center and adjust the settings. Incomplete transaction are generally the consequence of missing objects and/or links. The AIP platform provides various tools that can be used to complement the application analysis by creating the required elements.

For more information on transaction validation, refer to the section: Transaction Function Completeness.

Calibration

Ignore delete elements

The first step of the calibration is to ensure the boundary of the application. The output of the Transaction Review meeting provides an exhaustive list of transactions implemented in the application.

All transactions that have not been validated by the SME or that are considered as being technical coding should be excluded from the computation. Rules associated to these exclusions will be saved in the calibration kit.

For more information on calibration, refer to the section: Transaction Calibration.

Group and split elements

The second step of the calibration is to review the Transactional Functions and the Data Functions in order to verify if some of them should be grouped or split. This operation will complement the predefined processing done automatically on transactions.

Note that this review is the particularly important in a context of alignment with manual counting. The list of elements that have been grouped or split will be shared with a Certified Function Point Specialist (CFPS) in order to be validated.

For more information on calibration, refer to the section: Transaction Calibration.

Adjust type and FP values

The last step of the calibration is to adjust some values that have been calculated automatically. In this step the AI Administrator reviews the different types (EIF or ILF for Data Functions and EI, EO, EQ for Transactional Functions). He can also adjust the number of FP that have been calculated for the Functions.

For more information on how types and FP values can be adjusted, refer to the section: Transaction Calibration.

Calibration kit

This kit should contain the exhaustive list of Transactions, Data Entities, elements that have been ignored (such as lookup tables), specific generic rules and actions that have been performed. This list will allow the later reproduction of the same calibration for this application.

Manual counting alignment and functional calibration

In case the CAST Unadjusted Function Point count will be compared to a manual counting (usually performed by an independent organization), an additional calibration operation based on the the manual counting settings can be done by the AI Administrator.

Once calibration is completed, a baseline snapshot should be produced.

Automation

Once stabilized, Function Point configuration and calibration settings can be automated in order to be applied each time the functional size of the application will be measured.

For more information on this operation, refer to the section: 2.3. Analysis Automation.