1.3. Application Qualification

On this page:

Overview

As a pre-condition of a CAST AIP analysis and in order to qualify the application from a technical and value perspective, CAST recommends gathering high level technical and non-technical information to help qualify the target application for the purpose of the analysis. The technical qualification will be used to establish what **level of** "out of the box" support CAST has for the application, identify any show-stoppers such as the use of **unsupported technologies or frameworks** and **any potential customization** that may be required to accommodate exceptions.


When unsupported technologies are part of the application, you must assess whether the remaining part using the unsupported technologies is worth analyzing. CAST recommends accepting the application for analysis only when at least 60% of the business logic is supported.

The questions are primarily focused on identifying:

- Main technologies in use, including versions
- Estimated size of the application
- Architectural questions about cross technology architecture, functional domains, etc
- Technology specific questions giving insight into specific technical aspects and use of particular frameworks
- Main contacts within the application team

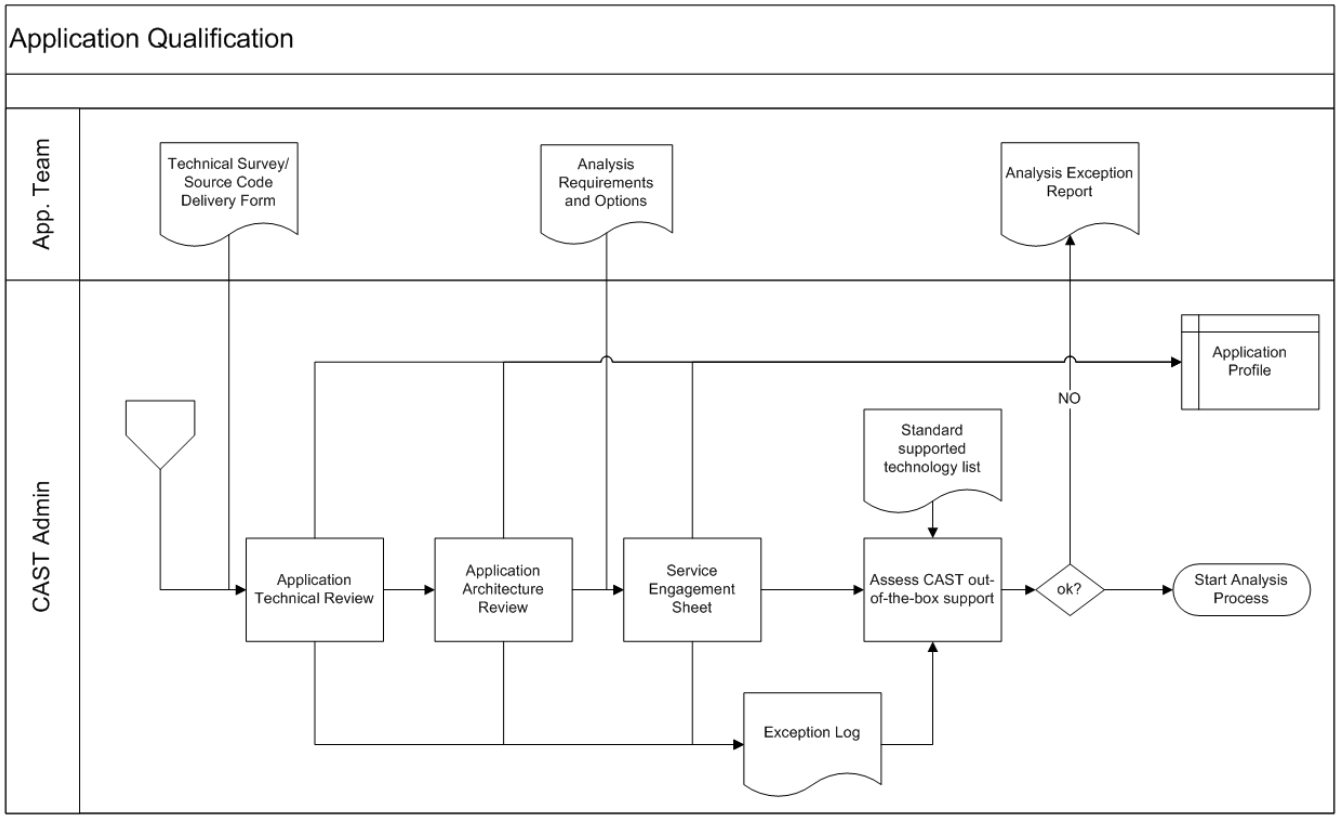
The non-technical qualification will help to establish what level of potential value could be derived by analyzing the application. Questions are primarily focused on identifying:

- Business related information e.g. criticality, number of users, major releases, etc.
- Application information e.g. maturity level, package solutions in use, such as PeopleSoft, etc
- Team organization e.g. maturity level of dev team, in house/out sourced dev, staff turnover, etc
- Quality e.g. production downtime hours, number and type of support calls, known issues, etc.

Tangible outcome of this step includes:

- Application Technical Survey and Source Code Delivery form
- Architecture review diagram and notes
- Service Engagement Sheet
- Analysis Exception Form (if the application is disqualified for technical, schedule or benefits vs panned-effort reasons and the analysis request rejected by the AI Center)

The following image presents this in more detail:



The key tasks depicted above include:

- Recommended approach
- Technology specific information
- Code pre-processing

The execution of tasks 1, 2 and 3 is usually combined. A single review meeting is scheduled to gather and validate all required information.

Application technical review

Gather information about the application's technological composition and structure. For example, for a typical **J2EE application** you should determine:

- The JDK version used to compile the source code
- The frameworks used and the version numbers
- The different application layers
- The archives (third-party)
- The application server used to run the application

This information will support the "out of the box" analysis check and will help ensure that during the source code delivery validation you have access to all relevant application components. In the case of a J2EE application, this means ensuring the App Team has delivered, for example:

- All the JAVA/JSP application source code, the relevant XML and .properties files and all the client files (HTML, JavaScript ...).
- The deployment descriptor files required for EJB and Web Services analyses
- The archives of libraries required to run the application, including those of the application server

As an example, in the case of a J2EE application, many of the standard configurations such as JDK version, use of standard framework etc. are automatically discovered by the CAST Delivery Manager Tool by parsing the project file and are included in the delivery package. Others options such as the JAVA/JSP file version, EJB version and web services need to be manually configured by the CAST Admin based on the information gathered in these forms or discovered by inspecting the delivered source code.

As such, the more thorough and complete the qualification step is, the least effort is required from the CAST Admin to discover the correct analysis settings on his/her own by examining the delivered source code.

Recommended approach

The recommended approach for executing this process step leverages the data collection form(s) identified above: forward the forms to application team stakeholders identified during the App. Team Kick-off for completion and follow-up with a meeting with all key application team stakeholders. The objectives of this meeting include:

- review any information provided (accuracy, clarity, completeness, etc.) by the app team before the meeting
- document in the form any relevant missing information

The Application technical review may include the discovery of the analysis purpose. Understanding the Application team's intended use of the analysis results can help guide and direct the CAST Admin in deciding weather "a good configuration is good enough". This may be the case for example when fine tuning transaction identification: depending on the specific analysis use case, it may be sufficient to identify perhaps less than 70% of the expected transactions where in other cases a more precise tune up of the configuration would be needed. Thus, an understanding of the analysis purpose could help optimize the analysis value vs. effort ratio.

Why is it important?

The information collected is used to:

- evaluate completeness of the delivered source code
- assess if an analysis with "out-of-the-box" support is possible and/or determine any potential solutions to address any limitations /constraints

Technology specific information

Please see the following pages for more information specific to each technology:

- .NET - application qualification specifics
- C and Cpp - application qualification specifics
- J2EE - application qualification specifics
- Mainframe - application qualification specifics
- ABAP - application qualification specifics
- Oracle Forms and Reports - application qualification specifics
- Business Objects - application qualification specifics

We have provided this as background information to ensure that the terminology used to describe elements in the source code is known by all.

Code pre-processing

Wherever possible, this step must be executed proactively to save time and improve analysis efficiency. Typically, the need for pre-processing may be triggered by the information gathered during qualification of the application with the application team. Some of the more common situations are listed below:

Creating participating databases from scripts

Because an SCM can only manage files, and as the database schema of an application is strictly dependent on the version of the application, the database schema description is normally stored in file format in the SCM with the application source code. The descriptions can be stored as simple DDL scripts, or in a proprietary format. On the other hand, CAST analyzes only databases on servers, not scripts, as such, you will need to "pre-process" these SCM-fetched database scripts to transform them into participating databases.

POSSIBLE SYMPTOM(s): SQL file extension found and no database package

Reorganize code

Sometimes the code inside each file is good for analysis, but the file organization is not satisfactory (this can happen for instance with web applications when URL-related pages are not folder-related using the same pattern, or when java source code is not in a folder path that matches their package). In these cases, you will probably need to do some pre-processing to move the files to a more appropriate location.

POSSIBLE SYMPTOM(s):

- Application qualification indicates that the development and deployment environments are different
- Cursory inspection of .war or .ear files hints to a different structure

Application architecture review

The Application architecture review aims to develop a high level understanding of the application architecture to help with the configuration of the analysis and the discovery of transactions. This meeting should include the Application Architect, Senior developer, etc. as appropriate.

Why is it important?

The information collected here is used to:

- evaluate completeness of the delivered source code
- ensure a complete and accurate analysis configuration
- support the configuration of transactions
- support Automated FP calculations

Analysis options

To completely define the analysis requirements, the CAST AI Admin needs to determine which analysis options should be considered during the analysis. This includes:

- escalated links
- XXL table rules
- User Input Security Dataflow
- User Defined Module (UDM) design (portfolio tree)
- Aggregation mode
- any additional application team expectations

Why is it important?

- document application team's expectations
- in a **M**anaged **A**nalytic **S**ervices (MAS) context, capture the terms of the analysis service contract between FO and BO
- essential for modules and configuration of other analysis options

Assess CAST out of the box support

Armed with the information gathered, the CAST AI Admin must validate out of the box support against the Supported Technologies. Also review the known Analysis Limitations.

Any technology, framework, etc. not explicitly mentioned in this list is not officially supported.

Solutions and or workarounds may exist in some cases that are supported by the CAST PS Team: these are considered out of scope and will not be addressed here.

When **unsupported technologies** are part of the application to be analyzed, an analysis exception report must be created and forwarded to the requester of the analysis. This will halt the analysis process.

Why is it important?

- identify any exceptions to out-of-the box support as early as possible
- improve efficiency and avoid reworks

Previous: 1.2. Delivery Managers and other key participants
Next: 1.4. Deliver the application source code version