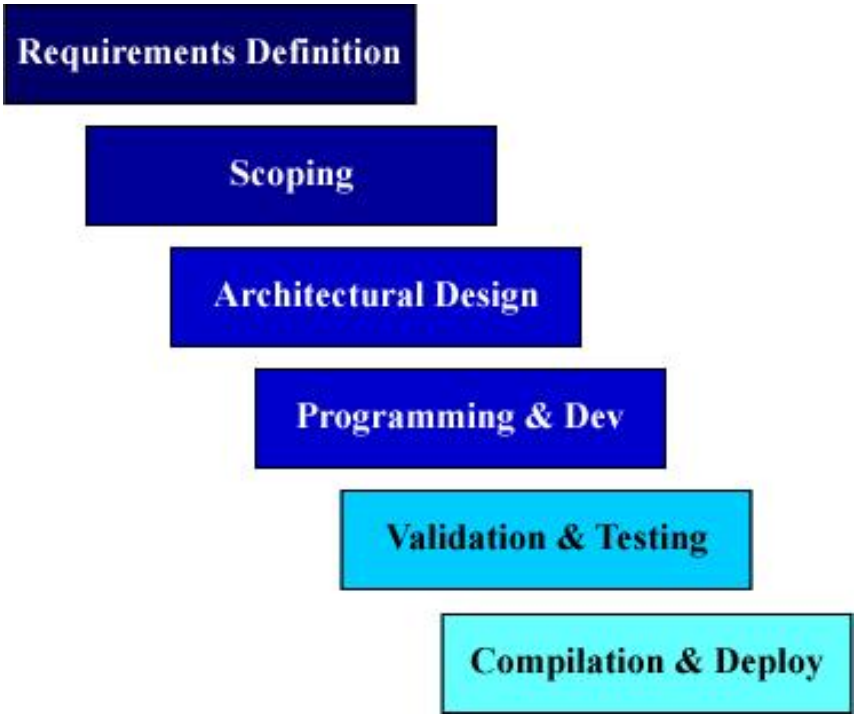
BELLATEQ
THE ART & SCIENCE OF TEQNOLOGY

SECTION 2 – PROGRAMMING & DEVELOPMENT

DEVELOPMENT METHODOLOGY – THE WATERFALL APPROACH



The Waterfall model of software development is a top-down, sequential approach to the design, development, testing and deployment of commercial-grade technology solutions.

The phases of the waterfall approach correspond to the phases of the software development lifecycle ("SDLC") which include – Requirements Definition, Enhancement Analysis and Corrective Analysis (collectively referred to as "Scoping"), Architectural Design, Programming and Development, Validation and Testing, Compilation and, finally, Deployment. Each of these phases is defined below.

REQUIREMENTS DEFINITION

This phase of the SDLC is a "big picture" exercise that establishes the point, purpose and goals for the next product release. Questions that need to be answered during the requirements definition phase include but are not limited to:

- Is this upcoming release primarily a bug fix release where we fine-tune existing system features and possibly introduce a few new capabilities or is

it a major functional release where we upgrade the user interface, introduce entirely new measures of functionality and seek to redefine the overall user experience while also addressing key algorithm errors and defects?

- Is this upcoming release targeted at our current customers with the goal of realizing deeper vertical market penetration or is it designed to enable the enterprise to attract customers in new markets thereby driving horizontal market expansion?

- Is this release expected to define (or further define) the enterprise as a thought leader or fast follower or is it expected to enable us to catch up to the competition and current functional standards and market expectations?

- What do our users expect this next release to do for them?

- What promises have we made and expectations have we set, as an organization, regarding this next product release?

ENHANCEMENT ANALYSIS

This phase of the SDLC is focused on what needs to be done to make the product more functionally capable. As opposed to deciding which bugs and defects will be addressed, the enhancement analysis phase seeks to answer the question "what will users be able to do with this next version of our software that they cannot do with our current software?" Examples of software fitness issues identified during the enhancement analysis phase include, but are not limited to:

- New or substantially modified user interface

- Support for new platforms (tablets, smart phones, additional operating systems, cloud computing, wireless services, etc.)

- Entirely new modules that expand the definition of "purpose for which the software is intended"

CORRECTIVE ANALYSIS

This phase of the SDLC is focused on existing features that may need to be improved and upgraded, known defects that need to be corrected and customer requests for the enhancement of currently available functionality. Functions and features of the current software release that require enhancement or programmatic correction drive the corrective analysis process. Examples of software fitness issues identified during the enhancement analysis phase include, but are not limited to:

- Fixing known defects that prevent users from engaging the software in the manner for which it was intended according to past Fitness-For-Use publications
- Expanding current functionality capacities
- Improving processing times and throughput
- Adding support for user compliance with new regulatory issues

These Scoping and Analysis phases of the SDLC conclude with the release of the Fitness-For-Use documentation ("FFU"). The FFU details the improvements to be included in the next release, the target market (and target users) for the next release and the key business case issues that the next release will address on behalf of the target users.

While the FFU details a technological commodity it is not a technical document. It should be written using business language that can be understood by end users and those who initiate, influence and approve purchasing decisions at the customer level.

The FFU is critical in that it enables the testing and quality control group to begin development of the master test plan while providing the sales & marketing group with critical information regarding product positioning, market penetration strategy and sales collateral content development.

ARCHITECTURAL DESIGN

Up until now, the phases of the software development lifecycle have focused on *what* the next software release is going to do. In the Architectural Design phase, the enterprise begins to define *how* the software is going to do it. What languages, programming platforms and hardware tools are going to be used to enable programmers and developers to complete the software development lifecycle?

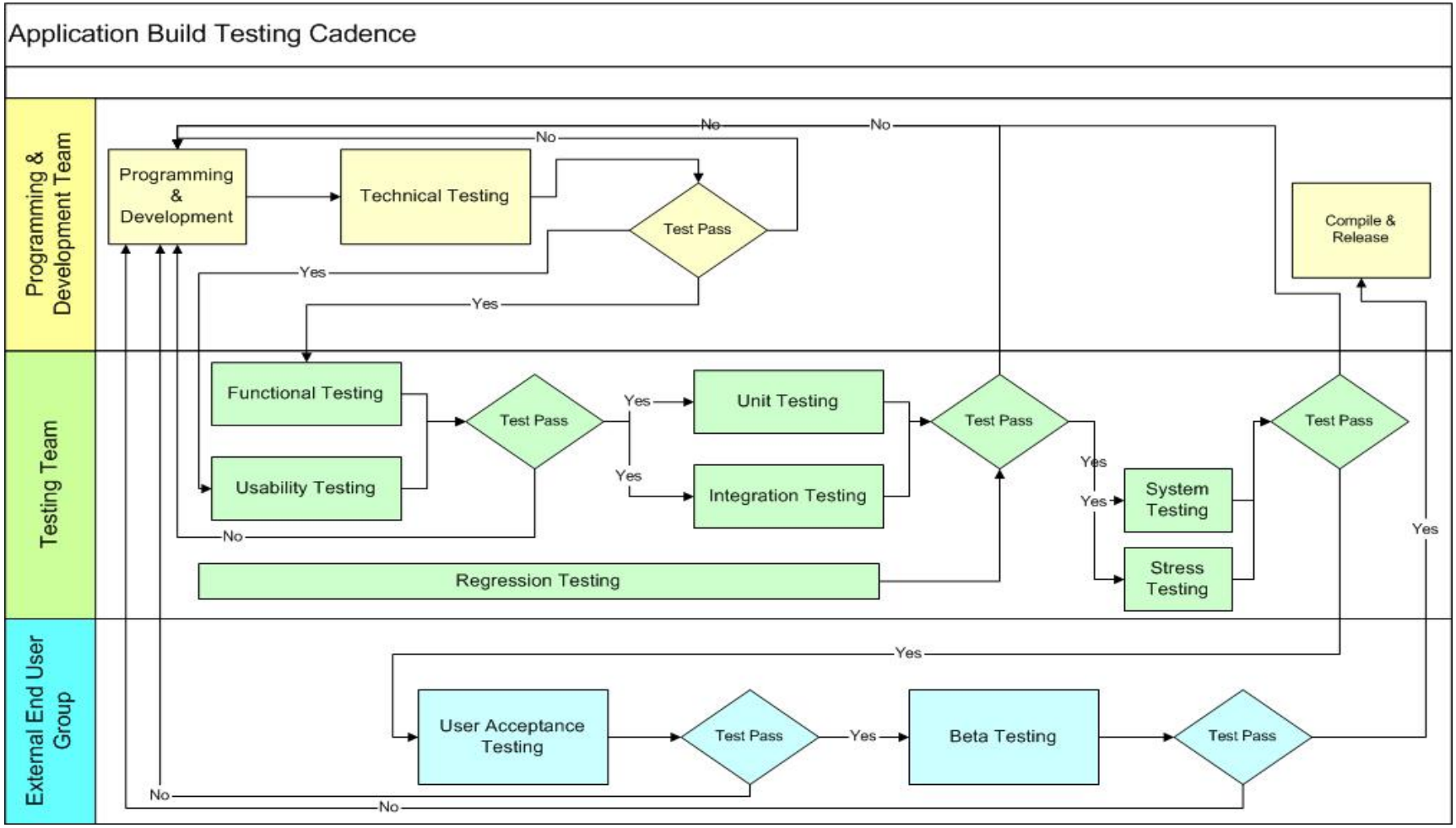
What approaches and programmatic designs are required to create software that meets the functional user expectations set in the Fitness-For-Use document? These questions must be answered in this phase of the SDLC.

The Architectural Design phase of the SDLC concludes with the release of the technical specifications documentation ("Tech Spec"). The tech spec is a technical document written *by* developers and architects *for* developers and architects. As such, it should include architectural design drawings, data flow maps, functional flow charts, underlying database table specifications and design standards for both the functional and user interface issues to be changed and/or introduced in the next release.

The Tech Spec is critical in that it keeps the all members of the development team on the same page even if they are not working on the same function, feature or module. A well-written tech spec goes a long way towards ensuring a coordinated development effort and a resultant deliverable that is consistent in design, construction and programmatic style.

With regards to the testing and quality control team, the technical specification document enables the test leader to begin defining the use cases, test scenarios and environmental configurations that will drive the tactical execution of the product testing lifecycle.

DEVELOPMENT & TESTING FLOW CHART



BELLATEQ
THE ART & SCIENCE OF TEQNOLOGY

DEVELOPMENT & TESTING NARRATIVE

The programming and development phase is beginning of software build and construction. It is during this phase of the SDLC that the programmers, architects, user interface designers, development leader and test leader all work together on the construction of the new software product.

The programming & development phase temporarily concludes with the release of the first testable version of the product. While functional and UI testing may have already started prior to completion of the programming & development phase, the release of this first testable version – the Alpha – represents the first opportunity the testing team has to execute end-to-end testing on all functions, features, enhancements and corrections documented and called for in the FFU.

With the commencement of end-to-end testing, the developers will re-start the programming and development phase of the SDLC with a focus on bug fixes and usability improvements found and reported by the testing team. This back-and-forth process continues until the testing team completes all functional, usability, unit, integration, system and regression testing. This phase of the SDLC is again temporarily completed with the release of the second testable version – the Beta1.

The release of the Beta1 signals the commencement of user acceptance testing by the quality control team. As in the first round of testing, developers again re-start this phase of the SDLC with a focus on the usability improvements documented by the testing team and approved by the test leadership group (Development Leader, Test Leader, Sales Leader, Marketing Leader).

With the completion of programming work on all reported usability improvement issues completed, the Programming & Development phase of the SDLC concluded and the Beta2 version of the software is released. (The Beta2 version is sometimes referred to as the “Pre-release” version.)

Upon release of the Beta2, it is expected that all features and functions detailed in the FFU are ready for pre-sales evaluation by external third parties. The Beta2 is the first iteration of the software that will be seen and used by individuals who are employed by customers of the enterprise as opposed to the enterprise itself. With the Beta2 in test-users hands, the Business Acceptance phase of the testing lifecycle begins.

By the time the enterprise is in Beta2 and external resources are testing and evaluating the software all defects should be corrected. Customer personnel should not be confirming that the products work. Business Acceptance Testing is

when external users confirm that the new products work *the way they expect it to work.* It is during Business Acceptance Testing that the enterprise confirms that the next planned release will at minimum *meet* user expectations.

COMPILATION

Programming is now complete and the software does everything the FFU says it is supposed to do. Testing is finished and the enterprise has confirmed that the software meets or exceeds customer expectation. The Compilation phase of the software development lifecycle now begins and is focused on preparing the application for distribution to and use by paying customers.

All distribution media (web download, CD installation, etc.) is prepared and tested. Collateral is required (labeled CD platters, users manuals, installation guides, user training and education collateral, etc.) is completed by the IT professional services team during this phase.

The Compilation phase concludes with:

- Notification to the sales department that orders can now be taken (or pre-orders fulfilled) for the new version of the software
- Notification to the marketing department that they can publish press releases announcing the commercial availability of the new version
- Customer service personnel notifying Upgrade & Enhancement contract subscribers that a new version of their software is ready for download and/or installation
- Communication by Sales and Sales Support personnel to customers that if they have budgeted for system upgrades, user training, on-site installation and/or consulting services related to the next version of their software that those services are now available for purchase and if they haven't budgeted for those issues they may wish to consider them going into their next budgeting cycle.

PROGRAMMING & DEVELOPMENT PROCESS DEFINITION

This process definition section includes detailed process flow charts and RACI diagrams for the key deliverables, as defined in the Programming and Development group at Cott Systems.

The flow charts in this section use graphic images that are defined in the tables below:

Action, Work Stream and Executed function Symbols

Symbol	Name	Description
	Process	Represent a Process or action step. This is the most common symbol in both process flowcharts and process maps.
	Predefined Process (Subroutine)	A Predefined Process symbol is a marker for another process step or series of process flow steps that are formally defined elsewhere. This shape commonly depicts sub-processes (or subroutines in programming flowcharts). If the sub-process is considered "known" but not actually defined in a process procedure, work instruction, or some other process flowchart or documentation, then it is best not to use this symbol since it implies a formally defined process.
	Alternate Process	This flowchart symbol is used when the process flow step is an alternate to the normal process step. Flow lines into an alternate process flow step are typically dashed.
	Delay	The Delay flowchart symbol depicts any waiting period that is part of a process. Delay shapes are common in process mapping and are often included in process flowcharts where periodic batch processes must be executed before the overall work stream can be completed.
	Preparation	Any process step that is a Preparation process flow step including but not limited to a set-up operation, user login, or user-defined parameter definition.
	Manual Operation	Manual Operations flowchart shapes show which process steps are not automated. In data processing flowcharts, this data flow shape indicates a looping operation along with a loop limit symbol.

Branching and Control of Flow Symbols

Symbol	Name	Description
	Flow Line	Flow line connectors show the direction that the process flows. Flow charts in North America typically flow from top to bottom and left to right.
	Terminator	Terminators show the start and stop points in a process. When used as a Start symbol, terminators depict a *trigger action* that sets the process flow into motion.
	Decision	Indicates a question, binomial response expectation or branch in the process flow. Typically, a Decision flowchart shape is used when there are 2 options (Yes/No, Go/No-Go, etc.)
	Connector (Inspection)	**Flowchart Context**: In flowcharts, this symbol is typically small and is used as a Connector to show a jump from one point in the process flow to another. Connectors are usually labeled with capital letters (A, B, AA) to show matching jump points. They are handy for avoiding flow lines that cross other shapes and flow lines. They are also handy for jumping to and from a sub-processes defined in an area other than the main flowchart. **Process Mapping Context**: In process maps, this symbol is full sized and shows an Inspection point in the process flow.
	Off-Page Connector	Off-Page Connector shows continuation of a process flowchart onto another page. When using them in conjunction with Connectors, it's best to differentiate the labels, e.g. use numbers for Off-Page Connectors and capital letters for Connectors.

Symbol	Name	Description
	Merge (Storage)	**Flowchart Context**: Shows the merging of and concatenation of multiple processes or process outputs into one piece of information or a single artifact. **Process Mapping Context**: commonly indicates storage of raw materials.
	Extract (Measurement)	**Flowchart Context:** Shows when a process splits into parallel paths. Also commonly indicates a Measurement, with a capital 'M' inside the symbol. **Process Mapping Context:** commonly indicates storage of finished goods.
	Or	The logical Or symbol shows when a process diverges - usually for more than 2 branches. When using this symbol, it is important to label the out-going flow lines to indicate the criteria that must be met in order for process flow to follow each branch.
	Summing Junction	The logical Summing Junction flowchart shape is shows when multiple branches converge into a single process.

Input and Output Symbols

Symbol	Name	Description
	Data (I/O)	The Data flowchart shape indicates inputs to and outputs from a process. As such, the shape is often referred to as an I/O shape.
	Document	Document flowchart symbol is for a process step that produces a document or artifact. Keep in mind that documents do not necessarily mean printed outputs.
	Display	Indicates a process step where information is displayed to a person (e.g., PC user, machine operator, customer, etc.)
	Manual Input	Manual Input flowchart shapes show process steps where the operator/ user is prompted for information that must be manually input into a system.

File and Information Storage Symbols

Symbol	Name	Description
	Stored Data	A general Data Storage flowchart shape used for any process step that stores.
	Database	This flowchart shape depicts a database consisting of columns corresponding to fields and rows corresponding to records.
	Direct Access Storage	Direct Access Storage | Hard Drive. Databases are usually stored on Direct Access Storage appliances.
	Internal Storage	Used in programming flowcharts to represent information stored in memory, as opposed to on a file. Arrays are the most common internally stored artifact but this symbol may also be used to represent cookies stored locally on a web applet users machine or device.

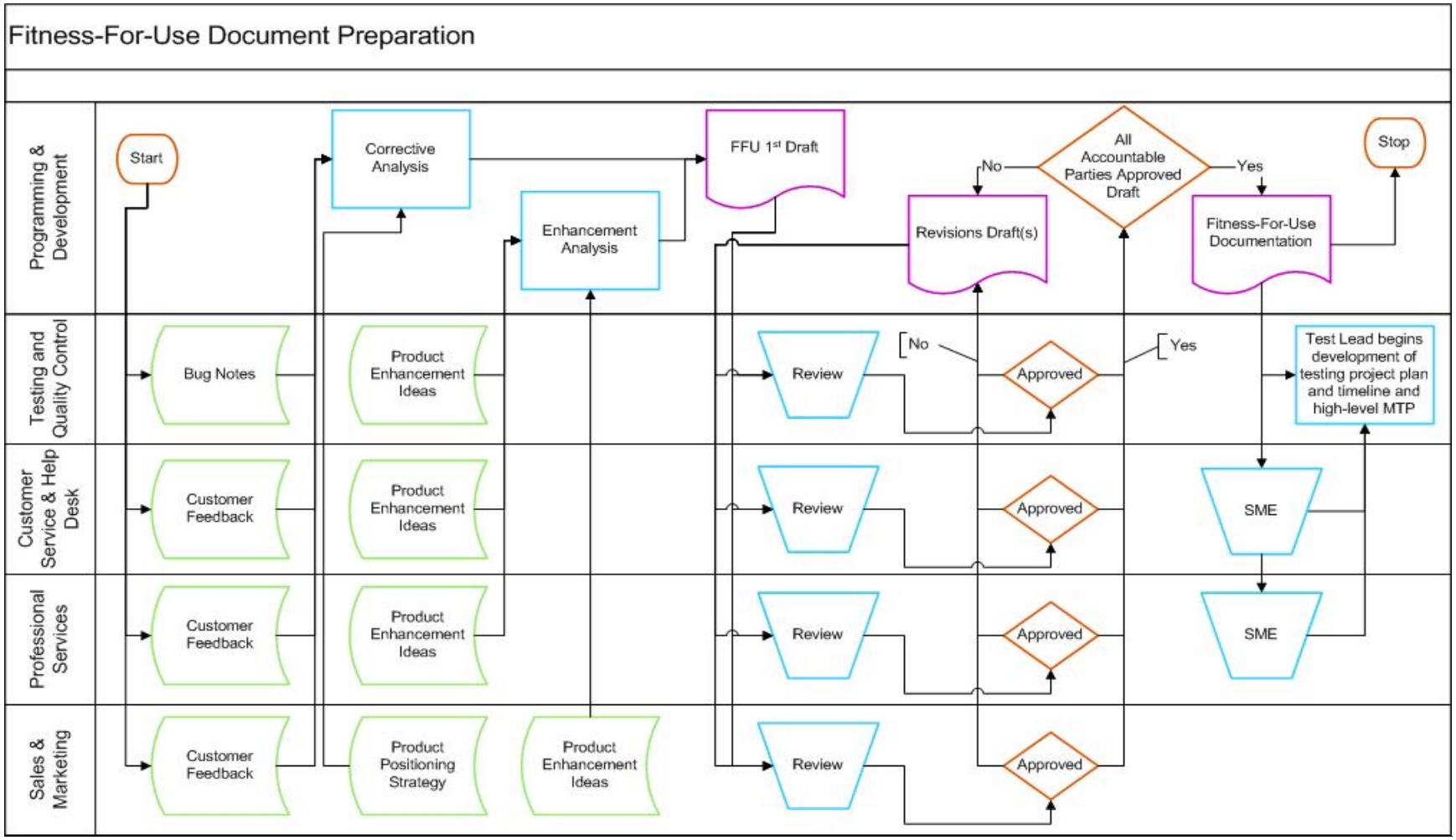
The RACI diagrams in this section use the following coding nomenclature:

SCENARIO	POTENTIAL RESOURCE ALLOCATION ISSUE
R – Responsible	Responsible parties own the project, work stream and/or deliverable assigned to them and are ultimately the point person for their quality and on-time delivery. Responsible parties, by definition are Accountable and must sign off on projects, work streams and deliverables before they can be considered complete
A - Accountable	Accountable parties must sign off on (approve) projects, work streams and or deliverables before than can be put into production or made available for external distribution
C – Consulted	Consulted parties have information, own content, have capabilities or are subject matter experts such that their contributions are necessary for successful completion of a project, work flow or deliverable
I – Informed	Informed parties must be notified of the results and/or successful completion of projects and work streams and should be made aware of the availability of key deliverables

PRODUCT FITNESS-FOR-USE RACI DIAGRAM

Stakeholder	**R**esponsible	**A**ccountable	**C**onsulted	**I**nformed
Development Team				
Development Lead				
Testing Team				
Test Lead				
Customer Service Team				
Customer Service Manager				
Sales Team				
Sales Manager				
Professional Services Team				
Professional Services Manager				
VP - Operations				
CFO				
CEO				

PRODUCT FITNESS-FOR-USE FLOW CHART

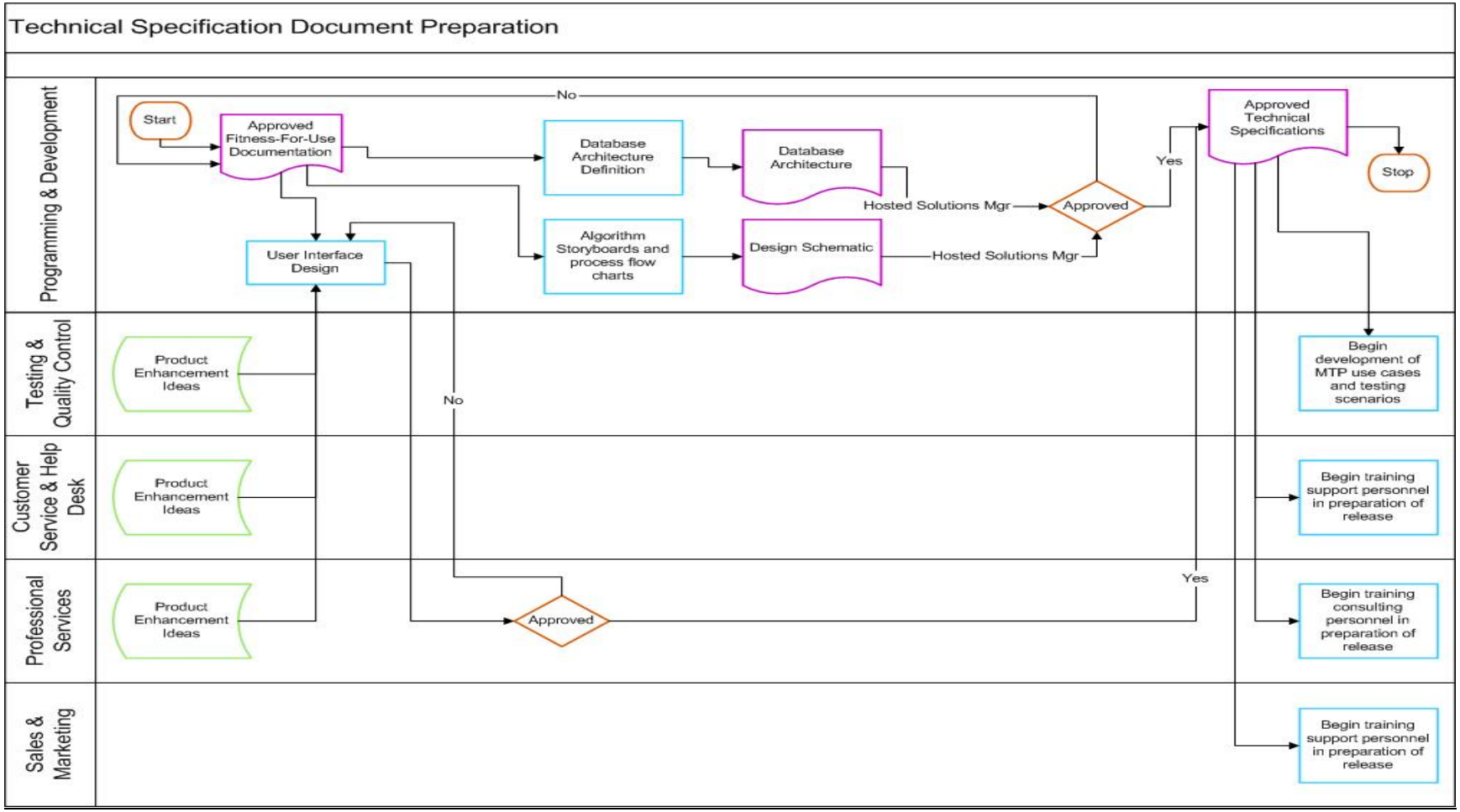


PRODUCT TECHNICAL SPECIFICATION RACI DIAGRAM

Stakeholder	Responsible	Accountable	Consulted	Informed
Development Team				
Development Lead				
Testing Team & Test Lead				
Hosted Solutions Manager				
Customer Service Team				
Customer Service Manager				
Sales Team				
Sales Manager				
Professional Services Team				
Professional Services Manager				
VP - Operations				
CFO				
CEO				

BELLATEQ
THE ART & SCIENCE OF TEQNOLOGY

PRODUCT TECHNICAL SPECIFICATION FLOW CHART



SOFTWARE DEVELOPMENT LIFECYCLE RACI DIAGRAM

SDLC Lifecycle Phase	Responsible	Accountable	Consulted	Informed
Requirements Definition	Dev Lead	Test Lead, Pro Srvcs Lead	Test Team, Cstmr Support Team, Sales Team, Mktg Team, Customers	Senior Exec Team
Enhancement Analysis	Dev Lead	Sales Lead, Mktg Lead		
Corrective Analysis	Dev Lead	Test Lead, Cstmr Support Lead		
Design & Architecture	Dev Lead		Dev Team, Hosted Solutions Lead	
Programming & Development	Dev Lead		Dev Team	
Validation & Testing	Test Lead	Dev Lead		Dev Team, Senior Exec Team
Compilation	Dev Lead		Hosted Solutions Lead	Sales Lead, Mktg Lead
Deployment	Pro Srvcs Lead	Dev Lead, Test Lead		Senior Exec Team, Customers

PROGRAMMING & DEVELOPMENT FLOW CHART