ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

PROJET DE 9 CRÉDITS PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE LOGICIEL

PAR
LUCIE ST-GERMAIN

IMPLANTATION DU CYCLE DE DÉVELOPPEMENT RATIONAL UNIFIED PROCESS CHEZ TECSYS INC.

MONTRÉAL, MARS 2001

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

- M. Michel Lavoie, directeur de mémoire
  Département de génie électrique à l'École de technologie supérieure

- M. Claude Laporte, professeur
  Département de génie électrique à l'École de technologie supérieure

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET UN PUBLIC

LE 5 FÉVRIER 2001

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

IMPLANTATION DU CYCLE DE DÉVELOPPEMENT RATIONAL UNIFIED PROCESS CHEZ TECSYS INC.

Lucie St-Germain

(Sommaire)

La division de recherche et développement (R&D) de Tecsys Inc. désire revoir son cycle de développement et implanter le processus défini par Rational Inc.. le "Processus Unifié"(RUP). Le projet a débuté par une revue des processus actuellement utilisés dans chacune des équipes de la division de R&D. Les processus actuels ont été documentés. Les forces et les faiblesses ont été identifiées. Ensuite une étude de RUP a été effectuée en suivant un programme de formation chez Rational.

Une analyse comparative du processus actuel et de RUP a été réalisée afin d'identifier les écarts entre les deux processus de développement. Lors de cet exercice. nous avons identifié les activités reliées à la gestion de projet et à la gestion des exigences comme étant les faiblesses majeures de la division.

Cette analyse nous a permis de proposer à la division R&D quelques stratégies d'implantation. Parmi les stratégies proposées. l'implantation de l'enchaînement «gestion des exigences» a été sélectionnée comme projet pilote. Donc. nous avons procédé à une comparaison détaillée entre les activités et les artefacts produits dans R&D avec ceux suggérés par RUP. Les étapes d'implantation ont été identifiées et une équipe a été formée afin de procéder à la première étape qui est de remplacer notre document d'exigences par le document de vision tel que proposé par RUP.

Suite au projet pilote. les priorités de la division ont changé. Nous avons donc cessé le projet d'amélioration des processus. Quelques mois après l'arrêt du projet. le vice-président de la division a été remplacé. Le nouveau vice-président a travaillé plusieurs années dans des domaines où les processus sont importants à cause du haut niveau de qualité requis. Il a rapidement identifié les faiblesses de la division de R&D et désire continuer l'effort d'amélioration des processus en lui accordant une grande importance.

IMPLANTATION DU CYCLE DE DÉVELOPPEMENT RATIONAL UNIFIED PROCESS CHEZ TECSYS INC.

Lucie St-Germain

(Abstract)

The Research & Development division of Tecsys wants to review their current processes and implement the engineering process the "Rational Unified Process" (RUP).

The first step was to proceed with an assessment of the current processes that currently exist within each division's team. The current processes were documented: the positive and negative aspects were identified.

We continued with a comparative analysis between RUP and the R&D current processes to determine the gap between both. This activity helped us identifying which areas need to be improved first. After completion, we have identified major gaps in the "Project Management" and the "Requirements Management" workflow.

Afterwards, a few strategies were suggested to the vice-president to proceed with the implementation of the RUP. The implementation of the requirement workflow has been accepted as being the pilot project. We continued with a more detailed comparative analysis between the activities from the requirement workflow and the activities perform in R&D. Subsequently, a team was put in place to implement the vision document from RUP to replace our business requirement document.

Once the pilot project was finished, the division priorities changed and the process improvement project was postponed. After a few months, the R&D vice-president was replaced and since the new vice-president has experienced in businesses where processes and quality are required, he has identified our weaknesses and wished to continue the process improvement project.

TABLE DES MATIÈRES

Page

LISTE DES ABRÉVIATIONS ET DES SIGLES

4G/L	4th Generation Language
DBA	Data Base Administrator
DMS	Distribution Management System
ERP	Entreprise resource planning
FMS	Financial Management System
OO	Orienté Objet
PVCS	Polytron Version Control Software
RUP	Rational Unified Process
R&D	Recherche et Développement
WMS	Warehouse Management System

INTRODUCTION

Tecsys Inc. est une firme d'environ 300 employés qui fait le développement, la mise en marché et l'implantation d'un système ERP pour le domaine de la distribution. Le système ERP comprend trois produits majeurs. Le premier, un système de distribution (DMS Distribution Management System), a été entièrement développé par Tecsys. Le second, un système de gestion d'entrepôt (WMS Warehouse Management System) est un produit qui a été acheté et intégré avec notre produit de base. Le troisième, un système de gestion comptable (FMS Financial Management System) dont Tecsys a acheté la compagnie. Tecsys met en marché deux versions majeures par année et six versions pour corriger les anomalies. La clientèle de cette compagnie est située à 90% aux États-Unis. Tecsys a deux grandes divisions : Une division pour la recherche et développement (R&D) et l'autre pour le service à la clientèle. Seule la division de R&D est impliquée pour ce projet d'implantation d'un nouveau processus de développement.

La division de R&D de Tecsys utilise depuis plusieurs années un cycle de développement en cascade (voir Annexe A) qui a été adapté à leur environnement de développement en 4G/L (langage Informix). Depuis deux ans, cette division a entrepris un développement en langage Java, sélectionné afin d'utiliser une technologie plus récente donnant les avantages d'un développement orienté objet et une plus grande flexibilité au niveau de l'interface usager.

En quelques années, la division de R&D a connu une forte croissance et réalise que son processus de développement doit être revu afin d'être adapté à de nouvelles réalités : changement de personnel, développement orienté objet, clients de grandes envergures, etc.

Une étude a été effectuée par un comité afin de déterminer un processus de développement commercial qui conviendrait à la division R&D et ce comité a choisi

Rational Unified Process (RUP). La division R&D désire implanter ce processus pour le développement de tous ses produits autant les produits développés en Informix qu'en Java.

Mon mandat pour ce projet est :

i. D'analyser les processus actuels afin d'en déterminer les forces et les faiblesses.
ii. D'étudier le processus d'ingénierie offert par Rational soit le "Processus Unifié"(RUP).
iii. Définir une stratégie d'implantation de RUP.
iv. Gérer un projet pilote d'implantation.
v. Produire des recommandations pour le déploiement des nouveaux processus à la grandeur de la division R&D.

CHAPITRE 1

CYCLE DE DÉVELOPPEMENT ACTUEL

Ce chapitre décrit la méthode utilisée afin de réviser et de documenter les processus actuellement utilisés dans chacune des équipes de la division de R&D.

1.1 Analyser les processus actuels

La première étape du projet consiste à analyser les processus actuels de la division de R&D. Cette division est composé de plusieurs équipes et chacune de ces équipes jouent un rôle à l'intérieur du processus. L'équipe de direction de produit est responsable du contenu tant fonctionnel que technologique des prochaines versions des produits de Tecsys. L'équipe de développement est responsable de la partie développement, c'est-à-dire l'analyse fonctionnelle, l'analyse organique, la programmation et les tests unitaires. L'équipe d'assurance qualité est responsable des tests d'intégrations. L'équipe de déploiement des versions est responsable de la traduction en Espagnol de la documentation pour les usagers, la documentation d'installation, la formation et la publication des informations sur les versions des produits.

Donc, nous avons mis en place quatre groupes de travail ayant le mandat d'analyser les processus actuels de chacune des équipes. La technique utilisée est de d'identifier, de documenter et de comprendre tous les processus existants, documentés ou non, utilisés par chacune des équipes.

Afin d'avoir une constance dans l'évaluation des processus, les groupes de travail ont suivi les même étapes :

i. Identifier et documenter les processus formels et informels de l'équipe.
ii. Identifier les règles d'affaires qui régissent les processus identifiés.
iii. Identifier les contraintes imposées sur le processus.
iv. Identifier les métriques actuellement utilisées.

v. Identifier les outils actuellement utilisés.

vi. Identifier les aspects positifs des processus actuels.

vii. Identifier les aspects négatifs des processus actuels. Identifier ce qui ne fonctionne pas et qui devrait être amélioré.

viii. Identifier les facteurs considérés critiques au succès de l'implantation de nouveaux processus.

Un rapport par équipe a été rédigé comprenant les résultats de l'évaluation. Les rapports produits par chacune des équipes se retrouvent en Annexe.

Annexe B : Rapport pour l'équipe de direction de produit

Annexe C : Rapport pour l'équipe de développement

Annexe D : Rapport pour l'équipe d'assurance qualité

Annexe E : Rapport pour l'équipe de déploiement des versions

Afin de bien comprendre le rôle de chacune des équipes à l'intérieur de la division R&D, les sections suivantes résument les responsabilités de l'équipe, les faiblesses dans leurs processus et les causes qui ont été identifiées.

1.2 Direction de Produits

1.2.1 Sommaire

L'équipe de direction de produit est responsable du contenu tant fonctionnel que technologique des prochaines versions des produits de Tecsys. Cette équipe analyse les besoins du marché afin de déterminer l'évolution de la gamme de produits. Par la suite, un plan qui s'étend sur un à cinq années de l'évolution de nos produits est proposé et approuvé par un comité. Le plan annuel devient la base des exigences du produit qui sont développées dans les deux prochaines versions. À partir de ce plan plusieurs documents d'exigences sont produits. Par la suite, ces documents sont transmis à

l'équipe de développement afin de produire les documents d'analyses fonctionnelles. L'équipe de direction de produits travaille en collaboration étroite avec l'équipe de développement afin de s'assurer que les exigences sont bien comprises. Cette équipe joue également le rôle d'assurance qualité afin de s'assurer que les standards de Tecsys sont bien suivis.

1.2.2 Résultat de l'évaluation

Lors de l'étude des processus actuels, cette équipe a identifié les faiblesses suivantes :

i. Les documents d'exigences ne sont pas toujours produits. Les exigences sont souvent discutées verbalement avec l'équipe de développement.
ii. Les standards au niveau de l'interface usager du produit ne sont pas toujours respectés et imposés.

Ces faiblesses sont dues en partie au fait que :

i. Les ressources sont souvent assignées au projet basé sur la disponibilité et non sur l'expertise.
ii. Il y a une contrainte budgétaire et un manque de ressources.
iii. Il n'y a pas de processus officiels et documentés.
iv. Il n'y a aucune activité d'assurance qualité mise en place afin de s'assurer que le processus est suivi.

1.3 Équipe de développement

1.3.1 Sommaire

L'équipe de développement est responsable de la production de l'analyse fonctionnelle, l'analyse organique, la programmation, les tests unitaires, l'aide en ligne et la traduction en français des produits.

À partir, soit des documents d'exigences écrits par l'équipe de direction de produits, si disponible, ou soit à partir des exigences transmises verbalement, un ou plusieurs analystes sont assignés au projet.

Le ou les analystes prennent connaissance du document d'exigences si disponible et discutent des solutions possibles avec l'équipe de direction de produits. Après que les deux équipes se sont entendues sur une solution, l'analyste rédige un document d'analyse fonctionnelle. Ce document indique les changements qui doivent être apportés au logiciel et à la base de données. Le document d'analyse doit être révisé par l'administrateur de la base de donnée (DBA) et par l'équipe de direction de produits. Un estimé des efforts pour passer à l'étape suivante, c'est-à-dire pour effectuer l'analyse organique, est produit. L'équipe de direction de produit doit approuver le document d'analyse avant de passer à l'étape suivante. Puisque cette équipe est également responsable pour le budget accordé à l'amélioration du produit, l'équipe de direction de produit doit approuver l'estimé produit par l'équipe de développement.

Lorsque le document d'analyse et l'estimé de l'effort ont été approuvés, l'analyse organique débute. L'analyse organique indique les changements à apporter aux écrans et les changements techniques à apporter aux programmes et fonctions. Un estimé de l'effort pour la programmation est produit. Le document d'analyse organique et l'estimé doivent être approuvés par l'équipe de direction.

Lorsque le document d'analyse organique est approuvé, le chef de service d'une des équipes de développement assigne des programmeurs au projet.

Lorsque la programmation est terminée, un programmeur senior effectue une revue de code. La revue de code est effectuée selon les standards de programmation définis dans la division de R&D. Suite à la revue de code, l'analyste responsable pour l'analyse fonctionnelle effectue des tests de fonctionnalités avant de transférer le projet à l'équipe d'assurance qualité.

Lorsque la programmation est terminée, l'équipe de traduction vérifie l'aide en ligne afin de s'assurer que la terminologie utilisée par l'analyste est conforme au glossaire. Le glossaire est un document contenant les termes reliés au domaine de la distribution qui doivent être utilisés à travers les produits de Tecsys.

1.3.2 Résultat de l'évaluation

Lors de l'étude des processus actuels, cette équipe a identifié les faiblesses suivantes :

i. Les analystes doivent souvent produire une analyse fonctionnelle sans le document d'exigences.
ii. Les documents d'analyse fonctionnelle et organique ne sont pas mis à jour lorsqu'il y a un changement dans la solution proposée.
iii. Les équipes des différents produits ne collaborent par toujours entre eux. Il y a une inconsistance dans les solutions proposées lorsqu'un changement affecte plusieurs produits.
iv. Les solutions doivent fréquemment être modifiées à chaque étape du processus de développement à cause d'un oubli dans l'étape précédente et les documents ne sont pas modifiés pour refléter ces changements.

Ces faiblesses sont dues en partie au fait que :

i. Il y a un manque de ressources.

ii. L'environnement de développement est instable. Ce qui engendre beaucoup de perte de temps.

iii. Les ressources ne sont pas impliquées suffisamment tôt dans le projet. Les ressources ne comprennent pas les besoins d'affaires.

iv. Le processus n'est pas toujours suivi.

1.4 Équipe d'assurance qualité

1.4.1 Sommaire

L'équipe d'assurance qualité est responsable des tests par projet et des tests d'intégration avant la mise en marché d'une version des produits.

Suite à l'analyse organique, le projet est présenté aux programmeurs. Au même moment certaines ressources de l'équipe d'assurance qualité participent à cette présentation. Suite à cette présentation, l'équipe d'assurance qualité prépare un plan de test. Lorsque le projet est prêt à être testé, les ressources de l'équipe d'assurance qualité effectuent les tests du projet en se servant du plan de test qu'ils ont préparé. Lorsqu'une anomalie est détectée, le détail est transmis à l'analyste. Ensuite, l'anomalie est corrigée par l'équipe de développement et le programme corrigé (ou la composante) est retransmis à l'équipe d'assurance qualité.

1.4.2 Résultat de l'évaluation

Lors de l'étude des processus actuels, cette équipe a identifié les faiblesses suivantes :

i. Les plans de test produits par les différents membres de l'équipe sont inconsistants.
ii. Les environnements pour effectuer les tests sont souvent non disponibles.
iii. Les plans de test ne sont pas complets.

Ces faiblesses sont dues en partie au fait que :

i. Le processus n'est pas clairement défini et documenté. Aucune standardisation dans la méthodologie des tests.
ii. Les analyses fonctionnelles et organiques ne sont pas toujours disponibles.
iii. L'équipe d'assurance qualité est impliquée très tard dans les projets. Ils ne sont pas impliqués au niveau des exigences et de l'analyse fonctionnelle.

1.5 Équipe de déploiement des versions

1.5.1 Sommaire

L'équipe de déploiement des versions est responsable de la documentation des usagers, la traduction de ces manuels, la documentation d'installation, préparation de la formation et de l'information sur les versions.

1.5.2 Résultats de l'évaluation

Lors de l'étude des processus actuels. cette équipe a identifié les faiblesses suivantes :

i. La production de la documentation retarde énormément le déploiement d'une version.

ii. La documentation n'est pas toujours adéquate.

Ces faiblesses sont dues en partie au fait que :

i. Manque de ressources dans cette équipe.

ii. Les documents d'analyses ne sont pas disponibles ou bien non mis à jour.

CHAPITRE 2

ÉTUDE DE "RATIONAL UNIFIED PROCESS"

Ce chapitre décrit le processus de génie logiciel Rational Unified Process (RUP). Le premier sous-chapitre analyse ce processus et les deux autres indiquent les points forts et faibles du processus. Les points faibles et forts du processus ont été identifiés selon les besoins de la division de R&D.

2.1 Analyser le processus de développement proposé par Rational

Rational Unified Process (RUP) est un processus de génie logiciel basé sur le cycle de développement Unified Process Development proposé par Grady Booch, Ivar Jacobson et James Rumbaugh décrit dans le livre The "Unified Software Development Process" qui a été publié en janvier 1999. (Voir Annexe F)

RUP est un processus de génie logiciel commercialisé par Rational Software. La documentation de RUP livré en format HTML peut être installée sur l'intranet de la compagnie ce qui rend la documentation des processus accessible à tous. Un engin de recherche est également fourni avec RUP afin de faciliter la recherche de l'information.

Ce processus est basé sur les six meilleures pratiques de développement proposées par Rational et qui sont :

i. *Développer le logiciel de façon itérative*

   L'approche itération s'inspire sur le modèle en spirale de Barry Boehm. Le concept étant de séparer le projet en mini-projet en sélectionnant pour chaque mini-projet un sous-ensemble réduit d'exigences et quelques risques. On effectue une analyse détaillée, la conception, la réalisation et les tests sur ce sous-ensemble. À la fin de cette itération on valide les résultats et on procède à la seconde itération.

ii. *Gérer les exigences*

Une gestion des exigences permet un meilleur contrôle sur les projets, une qualité du logiciel améliorée, des coûts et délais réduits ainsi qu'une meilleure communication à l'intérieur de l'équipe.

iii. *Utiliser une architecture à base de composantes*

Les activités de conception de RUP sont centrées sur la notion d'architecture modulaire où l'on doit identifier, concevoir, développer et tester les composantes puis les intégrer progressivement dans le système. Une composante peut être un module, une fonction ou même un sous-système.

iv. *Modéliser graphiquement le logiciel*

RUP suggère d'utiliser le plus possible les graphiques. Les graphiques permettent de mieux comprendre le problème et sa solution.

v. *Vérifier la qualité du processus et des produits*

Il y a deux aspects à la qualité lorsque l'on développe des logiciels. Le premier étant la qualité du produit développé et le second étant la qualité du processus utilisé pour construire le produit. Le processus inclut la qualité de tous les artefacts produits au cours du cycle de développement et les points de contrôle afin de valider cette qualité.

vi. *Contrôler les changements apportés au logiciel*

Il est important de contrôler et d'analyser l'impact de tout changement demandé en cours de développement. Cela implique de garder une trace de toutes les requêtes, de tous malentendus et de déterminer les artefacts affectés par un changement.

L'objectif d'un processus est de décrire qui fait quoi, comment et quand dans le but de produire un résultat (pouvant être un résultat intermédiaire ou final qui est dans notre cas un produit logiciel livrable). Rational utilise, pour décrire son processus, les concepts

de travailleurs, d'activités et les étapes d'activités, d'artefacts et d'enchaînements d'activités, lesquels sont décrits ci-dessous.

Un travailleur indiquant le «qui» fait référence au rôle qu'un individu doit tenir dans le cadre de son travail. Il définit les responsabilités d'un individu ou d'un groupe d'individus à l'intérieur du processus. La description d'un rôle ne représente pas la description de tâche reliée au titre de la ressource mais en définie la responsabilité. Une ressource peut jouer plusieurs rôles dans le processus.

Une activité, indiquant le «comment» définit une unité de travail qui doit être accomplie à l'intérieur du processus. L'activité doit être définie de façon à être utilisable dans une planification ou dans un suivi de projet.

Un artefact, indiquant le «quoi», est un élément d'information fabriqué, modifié ou utilisé par une activité. Un artefact peut être un document, un modèle de classe, un programme source ou un programme exécutable.

L'enchaînement d'activités, indiquant le «quand», est une suite d'activités produisant un résultat observable. L'enchaînement d'activités donne la séquence des activités, les ressources impliquées et les artefacts qui doivent être produits.

Le processus d'ingénierie RUP est composé de neuf enchaînements d'activités : six d'ingénieries du logiciel et trois pour le support aux équipes d'ingénieries :

i. Modélisation métier,
ii. Gestion des exigences,
iii. Analyse et Conception,
iv. Implémentation,
v. Tests,
vi. Déploiement,

vii. Direction de projet.
viii. Gestion de la configuration et des changements.
ix. Environnement.

Afin de bien comprendre RUP il est important de saisir l'interaction entre les neufs enchaînements d'activités. Chaque enchaînement d'activités décrit la séquence des activités incluses dans cet enchaînement d'activités particulier. Afin de définir le processus global. il est important de décrire la contribution de chacun des enchaînements d'activités à un instant donné du processus d'ingénierie. Pour ce faire. Rational a séparé le processus en quatre phases :

i. Phase d'inception comprenant l'étude de faisabilité. le plan d'affaire. l'étendu du projet. le plan de projet et l'identification des risques.
ii. Phase d'élaboration comprenant l'architecture du système. la définition des exigences. gestion des risques.
iii. Phase de construction comprenant le développement et les tests du produit logiciel.
iv. Phase de transition comprenant le déploiement du produit logiciel aux usagers.

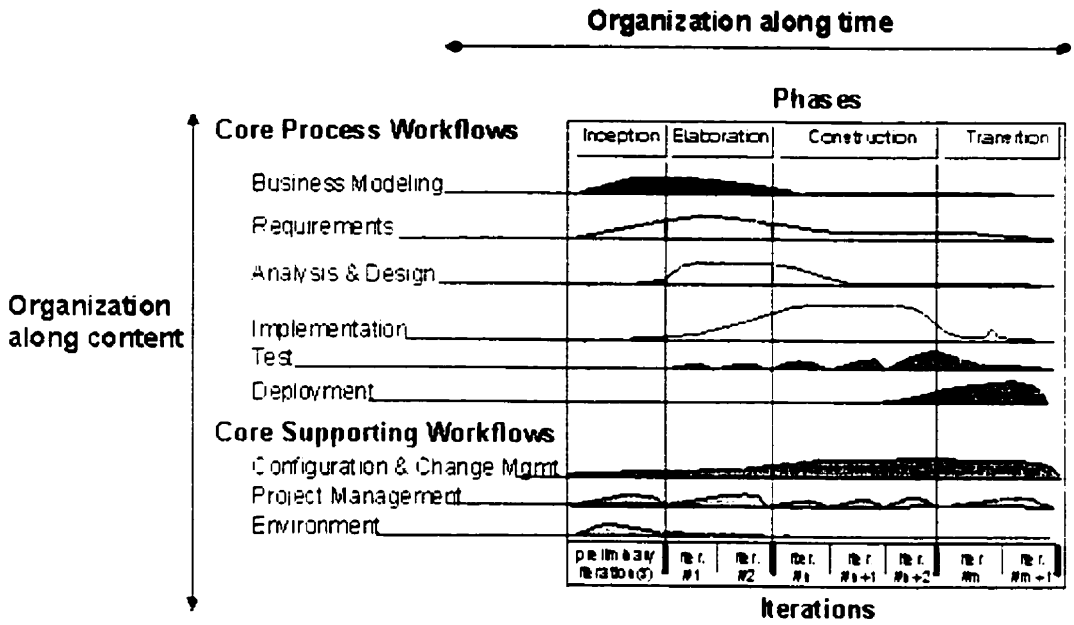


Figure 2.1 Processus d'ingénierie RUP [1]

À l'aide de cette figure il est facile de voir la contribution de chaque enchaînement d'activités à un instant donné du processus. Par exemple, lors de la phase d'inception, la modélisation métier et la gestion des exigences ont un rôle plus grand que les enchaînements d'activités d'analyse et de conception, de tests et du déploiement. Par contre, lors de la phase de transition on remarque que les enchaînements d'activités de modélisation de métier et de gestion des exigences ont moins d'impact dans le processus.

Cette figure illustre également les aspects dynamiques et statiques du processus RUP. L'axe horizontal représente le temps et montre l'aspect dynamique du processus en terme de cycle, phases, itérations et jalons. L'axe vertical représente l'aspect statique du processus en terme d'activités, artefacts, rôle et d'enchaînement d'activités.

[1] Cette figure a été tirée d'un document distribué sur le Web par Rational Inc.

2.2 Points Forts

Ce processus de développement met beaucoup l'accent sur :

i. La gestion du risque (en utilisant le concept d'itération).

ii. Le travail multidisciplinaire à l'intérieur d'un projet puisque à un temps donné d'un projet, des activités provenant des différents enchaînements d'activités ont lieues.

iii. Une bonne planification au départ du projet dans la phase d'inception. Une bonne planification est essentielle afin de déterminer les itérations qui seront les plus bénéfiques pour le projet.

iv. La documentation fournie avec RUP facilite l'apprentissage du processus. Elle est bien structurée car elle permet d'interroger le processus par enchaînement d'activités ou par rôle, par activité ou par artefact. Pour chacune des activités, des guides et des modèles de document sont fournis.

2.3 Points Faibles

Lors de l'évaluation de ce processus nous avons noté les points faibles suivants :

i. Les activités d'assurance qualité ne sont pas bien identifiées.

ii. La traduction de la documentation et du produit n'est pas couverte.

CHAPITRE 3

UTILISATION DE RUP DANS LA DIVISION DE R&D

Ce chapitre décrit comment Tecsys pourrait utiliser RUP pour le développement d'une nouvelle version de leurs produits.

Chacun des sous-chapitres explique une des phases de RUP. Les activités effectuées dans cette phase sont décrites dans un tableau en indiquant dans la dernière colonne si l'activité est effectuée dans le processus actuel de R&D. La lettre «O» indique que cette activité est effectu ée et satisfaisante. la lettre «N» indique que cette activité ne fait pas partie du cycle actuel et la lettre «P» indique que l'activité est effectuée partiellement ou de façon insatisfaisante.

Puisque Tecsys souhaite également utiliser les outils offerts par Rational. ce tableau en fait référence. Les outils de supports offerts sont Rose. Requisite-Pro et Clear-Quest. Rose est un outil de modélisation objet. Requisite-Pro est un outil permettant le suivi des exigences. il permet de suivre la traçabilité des documents à travers le cycle de développement. Clear-Quest permet de gérer les demandes de changement ainsi que les anomalies. Rational offre également le produit Clear-Case qui permet la gestion des versions. Actuellement la division de R&D utilise PVCS un outil équivalent. en conséquence nous ne ferons aucune référence à Clear-Case puisque le besoin est comblé par un outil équivalent.

3.1 Phase d'inception

La phase d'inception doit être réalisée avant le début du développement d'une nouvelle version des produits de Tecsys. Elle doit être commencée parallèlement à la phase de construction et de déploiement de la version précédente. Le but de l'inception n'est pas d'accumuler du code mais de fournir des réponses sur la faisabilité du projet.

Les objectifs de la phase d'inception sont :

i. Définir l'étendu des améliorations, changements à apporter dans la prochaine version.

ii. Déterminer les cas d'utilisations critiques c'est-à-dire les principaux scénarios qui influencent les choix de conception et les changements architecturaux au niveau des produits.

iii. Estimer l'effort global de développement et plus précisément de fournir des estimations détaillées des ressources nécessaires pour la phase d'élaboration.

iv. Identifier les risques potentiels (les sources d'incertitude).

À la fin de la phase d'inception, une décision au niveau du contenu et de la date de mise en marché de la prochaine version doit être faite.

Tableau 1

Les activités et artefacts de la phase d'inception

Activité	Artefacts	Commentaire	
Comprendre les besoins		Les besoins du marché et des usagers doivent être définis.	O
Définir la vision, la portée et les contraintes de la prochaine version des produits	Document de vision	Les documents de vision comprenant une liste de fonctionnalités doivent être produits. Le document de vision explique le contexte d'affaire du problème, les enjeux, les contraintes et l'impact sur les usagers.	P
		Les documents de vision doivent être intégrés dans Requisite-Pro. Ces documents sont le point de départ des exigences.	N
Définir le système	Modèles des cas d'utilisation	Les documents d'analyse fonctionnelle de haut niveau doivent être produits. Ces documents se basent sur les documents de visions et identifient tous les cas d'utilisation pouvant y	P

		être identifiés.	
		Les cas d'utilisation doivent être intégrés dans Requisite-Pro afin de retracer chacun des cas d'utilisation à une exigence définie dans un des documents de vision.	N
Réviser le modèle des cas d'utilisation		Les solutions proposées doivent être révisées et validées par l'équipe de direction de produits. Cette équipe doit s'assurer que les solutions proposées répondent aux besoins exprimés.	P
Préparer le plan de projet	Plan de projet	Un estimé de l'effort afin d'inclure tous les changements désirés dans la prochaine mise à jour doit être réalisé.	N
Déterminer les risques potentiels	Liste des risques	Les risques associés à la livraison de la version doivent être identifiés et évalués.	N
Réviser le plan de projet	Plan de projet révisé	Le plan de projet doit être revu par l'équipe de direction de produits afin de s'assurer que la date prévue pour la mise en marché de la prochaine version est respectée. S'il y a lieu les fonctionnalités et/ou la date prévue pour la mise en marché doivent être revues.	N
Accepter le plan de projet		La direction doit approuver la date de livraison et les livrables prévus pour la prochaine version avant de passer à l'étape suivante.	N

3.2 Phase d'élaboration

La phase d'élaboration a comme but de planifier les activités et les ressources, de spécifier les fonctionnalités et d'identifier, s'il y a lieu, les changements architecturaux.

Lors de la phase d'élaboration, les itérations de développement sont définies en considérant que chaque itération doit produire un système fonctionnel. Les itérations doivent être définies en considérant les risques. Les cas d'utilisation à risques élevés doivent être analysés, conceptualisés et développés en premier.

Les objectifs de la phase d'élaboration sont :

i. Analyser en détail les cas d'utilisation.

ii. Produire la conception.

Les activités de la phase d'élaboration doivent être réalisées par itérations. C'est-à-dire que les cas d'utilisations inclus dans la première itération doivent être détaillés avant la phase de la construction et lorsque les cas d'utilisation de la première itération sont en développements, l'équipe travaille à détailler les cas d'utilisation de la seconde itération, etc.

Tableau 2

Les activités et artefacts de la phase d'élaboration

Activité	Artefacts produits	Commentaires	
Raffiner le plan de projet	Plan de projet	Le projet doit être découpé en itérations en considérant que chacune des itérations doit produire une version livrable. Si dans le cas qu'il y ait dépassement dans une itération, il serait possible de mettre en marché une version fonctionnelle à la date prévue avec un peu moins de fonctionnalité. Le plan de projet doit prendre en considération ces itérations.	N
Réviser les demandes de changement		Les demandes de changement aux exigences doivent être révisées afin de déterminer l'impact sur l'estimé et la livraison. Suite à l'analyse de l'impact, les changements acceptés doivent être prévus dans une des itérations du projet.	N
Décrire en détail les cas d'utilisation	Cas d'utilisation	Selon les risques et les itérations du projet, les cas d'utilisation doivent être détaillés. Nous devons décrire en détail les cas d'utilisation qui engendrent des changements majeurs dans l'architecture du système afin de prendre les décisions au niveau architectural.	P
Prototyper l'interface utilisateur	Prototype de l'interface utilisateur	Si l'interface utilisateur est modifiée alors l'analyste doit produire un prototype (même papier) de l'interface utilisateur.	P

Réviser les cas d'utilisation	Cas d'utilisation	L'équipe de direction de produit révise les détails des cas d'utilisations.	P
		Les détails des cas d'utilisation doivent être intégrés dans Requisite-Pro afin de retracer les cas d'utilisation au modèle des cas d'utilisation.	N
Mettre à jour le glossaire	Glossaire des termes	Le glossaire des termes doit être mis à jour tout au long du projet afin de s'assurer d'utiliser la même terminologie.	P
Écrire les exigences non fonctionnelles	Exigences non fonctionnelles	Les exigences non fonctionnelles doivent également être décrites. Les exigences non fonctionnelles contiennent les contraintes associées au changement apporté aux produits de Tecsys.	P
Revoir la liste des risques	Liste des risques révisée.	Les risques doivent être évalués tout au long du projet afin de prendre des décisions efficaces.	N
Écrire les plans de tests	Plan de test	À partir des cas d'utilisation, l'équipe responsable pour les tests commence à rédiger les plans de tests.	P
		Les plans de tests doivent être intégrés dans Requisite-Pro afin de les retracer à chacun des cas d'utilisation.	N
Écrire la documentation des usagers	Documentation des usagers	À partir des cas d'utilisation, l'équipe responsable de la documentation des usagers commence à rédiger la documentation des usagers.	P
Réviser le plan de projet	Plan de projet révisé	Suite à l'activité afin de décrire les cas d'utilisation, le plan de projet doit être révisé afin de refléter tous changements.	N
Conception	Document de conception	À partir des cas d'utilisation (ou d'une partie), un document de conception est produit. Ce document de conception donne : • Pour le développement en 4G/L, le détail des modifications à apporter aux librairies, aux fonctions, aux programmes et à la base de données. • Pour le développement en Java, le détail des modifications à apporter aux classes, aux méthodes et à la base de données.	P
		Utilisation de Rational Rose pour la	O

		modélisation	
Inspecter la conception	Document de conception révisé	Un architecte senior doit réviser la conception suggérée. Un architecte de base de données doit réviser le modèle de données suggéré.	P
		Intégrer le document de conception dans Requisite-Pro afin de retracer les changements à faire aux cas d'utilisation.	N

3.3 Phase de construction

Le but de cette phase est de construire le système. Les cas d'utilisation sont développés et les tests réalisés.

Les objectifs de la phase de construction sont :

i. Obtenir un produit de qualité satisfaisante selon les critères définis par l'équipe responsable des tests des produits.

ii. Mettre au point rapidement des versions utiles (alpha, bêta et version finale)

Tableau 3

Les activités et artefacts de la phase de construction

Activité	Artefacts produits	Commentaires	
Créer l'espace de travail	Espace de travail	Un espace de travail doit être créé pour la première itération du projet.	O
Développer le code (Implémenter les classes)	Programme fonctionnel Classe fonctionnelle	En 4G/L les développeurs modifient les programmes, les librairies selon le document de conception en suivant les standards de programmation. En Java les développeurs implémentent les classes.	O
Effectuer les tests unitaires		Tests unitaires	P

		Les défectuosités doivent être inscrites dans Clear Quest.	N
Corriger les défectuosités		Les programmes ou les objets doivent être corrigés.	O
Intégrer les changements dans le produit	Produits intégrés	Les changements doivent être intégrés de façon incrémentale dans le produit	O
Effectuer les tests d'intégration		Les tests d'intégrations doivent être effectués.	O
		Les défectuosités trouvées doivent être inscrites dans ClearQuest.	P
Effectuer les tests de performance			N
Exécuter un test système		Lorsque les changements sont tous intégrés aux produits, un test du système doit être exécuté.	O
Évaluer l'itération	Plan de projet révisé	Le plan de projet doit être révisé à la fin de chacun des itérations afin de comparer l'effort, les délais et le contenu réels de l'itération à celui prévu initialement. Cette évaluation permet de réviser le plan de projet pour la prochaine itération.	N

Lorsque tous les changements associés à une itération sont intégrés dans le produit alors l'équipe d'assurance qualité effectue les tests de système.

3.4 Phase de transition

Le but de cette phase est de livrer le produit aux clients. Les clients de R&D étant les autres divisions de Tecsys : marketing, vente, équipe d'installation, équipe de support à la clientèle et division de Services.

La phase de transition démarre lorsqu'une version de nos produits est prête à être livrées à nos clients. L'objectif de cette phase est de livrer une version finale de nos

produits, incluant le logiciel et la documentation, aux autres divisions et départements de Tecsys et à nos clients.

Tableau 4

Les activités et artefacts de la phase de transition

Activité	Artefacts produits	Commentaires	
Définir les produits de référence	Produits de référence	Lorsque le produit est prêt à être livré, alors le responsable pour la gestion de configuration doit définir la nouvelle version des produits comme produits de référence.	O
Livrer les produits	Produit	Les produits ainsi que la documentation sont livrés aux utilisateurs.	O

CHAPITRE 4

COMPARAISON ET PLAN D'IMPLANTATION

Premièrement. ce chapitre décrit les activités de RUP qui ne sont pas utilisées dans R&D et qui devraient l'être. Deuxièmement. les activités effectuées dans R&D qui ne sont pas couverts dans RUP sont listées. Troisièmement nous expliquons le plan d'implantation proposé à R&D et le déroulement du projet pilote.

4.1 Description des activités RUP non utilisées dans la division R&D

4.1.1 Phase d'inception

i. *Définir la vision. la portée et les contraintes de la prochaine version des produits*
L'activité. afin de définir la vision. la portée et les contraintes de la prochaine version des produits. est présentement faite par la division R&D sans tenir compte des risques et de l'effort global avant d'entreprendre le développement. Une problématique auquel la division R&D doit faire face est la diversité des produits. Les équipes de développement sont séparées par produit et chacune des équipes fait un plan de projet en considérant seulement les changements qui affectent son produit. Aucun plan de projet consolidé au niveau de la division est produit.

ii. *Définir le système*
Il est important d'introduire la notion de cas d'utilisation afin de nous permettre de produire une analyse fonctionnelle de haut niveau. Une analyse de haut niveau doit être faite avant de procéder à une analyse exhaustive afin de déterminer rapidement l'étendu du travail à accomplir.

iii. *Réviser le modèle des cas d'utilisation*
Il est important d'introduire cette activité tôt dans le processus afin de réviser les solutions proposées et ainsi éviter d'approfondir une solution inadéquate.

iv. *Préparer le plan de projet*

Il est important d'introduire la notion de gestion de projet global pour tous les changements à apporter dans une nouvelle version de nos produits. Un estimé doit être produit à partir de l'analyse de haut niveau afin de déterminer s'il y a un écart entre l'effort et le budget prévu par l'équipe de direction de produit et ainsi apporter les correctifs à la solution ou au budget rapidement au début du projet.

v. *Déterminer les risques potentiels*

Cette activité doit être introduite dans la division de R&D afin d'être conscient des problèmes potentiels et de permettre la mise en place de plans de contingence.

vi. *Réviser le plan de projet*

Lorsqu'un plan de projet consolidé est disponible, la direction doit accepter la date de livraison et les livrables prévus pour la prochaine version avant de passer à l'étape suivante.

4.1.2 Phase d'élaboration

i. *Raffiner le plan de projet*

Présentement, il n'y a pas de plan de projet au niveau des modifications à apporter à la prochaine version des produits de Tecsys. Seul un plan d'affectation de tâches est fait et suivi par les chefs d'équipes.

ii. *Réviser les demandes de changements*

Présentement l'impact des demandes de changements n'est pas considéré et les documents produits ne sont pas mis à jour afin de refléter les changements.

iii. *Détailler les cas d'utilisation*

Présentement une analyse fonctionnelle détaillée est produite à partir du document des exigences fourni par l'équipe de direction de produits (si le document existe). La technique des cas d'utilisation n'est pas utilisée.

iv. *Mettre à jour le glossaire*

Notre glossaire est mis à jour plus tard dans le processus. L'équipe de documentation révise le document de conception et met à jour le glossaire ou demande à l'analyste de modifier certains termes. Cette activité devrait avoir lieu plus tôt dans le processus.

v. *Revoir la liste des risques*

La gestion des risques doit être introduite.

vi. *Écrire les plans de tests*

L'équipe d'assurance qualité écrit les plans de tests à partir des documents de conception et non à partir de l'analyse fonctionnelle. Ceci est dû au fait qu'il y a souvent des changements à la solution entre le document d'analyse fonctionnelle et de conception. Puisque les changements ne sont pas documentés, les plans de tests sont erronés.

vii. *Écrire la documentation des usagers*

Puisque qu'il y a souvent des changements à la solution entre le document d'analyse fonctionnelle, la conception et la solution programmée, la documentation des usagers est mise à jour lorsque la programmation est terminée.

viii. *Traçabilité des documents*

Présentement il n'y a aucune traçabilité entre les documents générés dans R&D. Régulièrement les oublis sont découverts lors des tests de système.

4.1.3 Phase de construction

i. *Effectuer les tests unitaires*

Actuellement le programmeur effectue des tests unitaires mais de façon informelle. Des critères de qualité auquel le programmeur devrait se référer n'ont pas encore été identifiés.

ii. *Inscrire les défectuosités dans un système approprié*

Actuellement les défectuosités sont gérées avec un logiciel de tableur (Excel). Cet outil n'est pas approprié et nous constatons une perte de temps et d'informations.

4.1.4 Phase de transition

Toutes les activités suggérées dans RUP pour la phase de transition sont effectuées dans la division de R&D.

4.2 Activités non couvertes dans RUP

Ce sous-chapitre décrit les activités effectuées dans R&D qui ne sont pas couvertes dans RUP.

i. *Traduction des produits*

Nos produits et la documentation des usagers doivent être traduits en plusieurs langues. L'activité de traduction doit être effectuée lors de la phase d'élaboration lorsque les documents d'analyses sont disponibles. La traduction de nos produits impacte également la phase de transition. La première livraison de nos produits se fait en anglais seulement. Les versions française et espagnole sont livrées quelques semaines et parfois quelques mois plus tard.

ii. *Révision de code*

Lorsque le temps le permet, le code écrit par le développeur est révisé. Les erreurs trouvées sont écrites dans un document de type Word.

4.3 Plan d'implantation et projet pilote

Plusieurs stratégies d'implantation de ce processus ont été discutées chez Tecsys. La première stratégie est de sélectionner un projet pilote utilisant le développement orienté objet en Java et de suivre le processus RUP pour ce projet. Cette stratégie d'implantation n'a pas été retenue pour les raisons suivantes :

i. L'équipe du projet doit être formée sur le processus au complet
ii. Ce projet pilote ne convainc pas Tecsys que le processus peut être adapté pour un projet de type traditionnel (développement en Informix).

La deuxième stratégie est de sélectionner un projet pilote utilisant le développement 4G/L et de suivre le processus complet pour ce projet. Cette stratégie d'implantation n'a pas été retenue pour les raisons suivantes :

i. L'équipe du projet doit être formée sur le processus complet
ii. Tous les gabarits de RUP doivent être revus et adaptés au développement de type 4G/L
iii. Beaucoup d'investissement pour un projet pilote donc le retour sur l'investissement est difficile à justifier

La troisième stratégie est d'implanter le processus par phase en commençant par la phase d'inception. Cette stratégie d'implantation n'a pas été retenue pour les raisons suivantes :

i. Le concept de chargé de projet au niveau d'une version à développer n'est pas présent dans R&D et demande un effort de changement majeur.
ii. La direction en place ne croit pas en une gestion de risques.

La quatrième stratégie est d'implanter le processus par enchaînement d'activités en apportant les changements nécessaires afin de l'adapter à Tecsys. Nous avons suggéré

de commencer par l'enchaînement d'activités de gestion de projet puisque suite à l'analyse exposée au chapitre 3, nous avons identifié la gestion de projet comme une de nos faiblesses majeures. La stratégie d'implanter par enchaînement d'activités a été acceptée mais nous ne pouvions pas entreprendre les changements au niveau de la gestion de projet. La raison étant que le rôle de chargé de projet dans R&D n'est pas défini et un tel rôle a un impact important dans la division au niveau des rôles et responsabilités de chaque équipe. La définition d'un tel rôle demande l'implication d'un niveau exécutif supérieur et à ce moment le niveau exécutif nécessaire (Vice-Président) n'était pas d'accord pour entreprendre un tel changement.

Donc, nous avons suggéré à la division de R&D de débuter par l'enchaînement d'activités de gestion des exigences puisque lors de l'analyse des processus actuels, la gestion des exigences est également une de nos faiblesses majeures. Cette stratégie a été acceptée par la direction. (Voir Annexe G pour une copie de la présentation)

Nous avons donc procédé à une comparaison détaillée entre l'enchaînement d'activités des gestions des exigences tel que proposé par RUP et notre processus actuel afin de déterminer l'effort d'implantation ainsi que l'effort de personnalisation de la documentation de RUP. (Voir Annexe H)

Plusieurs étapes d'implantation ont été identifiées :

i. Personnaliser et implanter le document de vision.
ii. Personnaliser et implanter le document des spécifications d'exigences.
iii. Modifier RUP afin d'utiliser la terminologie de Tecsys. Bien que Tecsys ait décidé d'adopter la terminologie RUP, il y a certains rôles dans RUP qui ne correspondent pas aux responsabilités de nos ressources.

Le processus de personnaliser et d'implanter le document de vision a été sélectionné comme étant notre projet pilote. À la fin de cet exercice une décision au

niveau de la haute direction doit être prise afin de déterminer si Tecsys continue le projet d'implantation de RUP.

Une équipe de travail a été formée afin de réviser et adapter le document de vision suggéré par RUP. Le document de vision remplace le document d'exigences qui est produit par l'équipe de direction de produit car il est beaucoup plus complet et explicite. Il explique le contexte d'affaire du problème, les enjeux et l'impact sur les usagers afin d'introduire au niveau des analystes une meilleure compréhension du domaine de la distribution (Voir Annexe I). La documentation de RUP a été adaptée afin de refléter les changements apportés au document de vision. Le rôle de directeur de produit a dû être ajouté puisque ce rôle est inexistant dans RUP.

Le nouveau gabarit pour le document de vision a été déployé à travers la division et le changement a bien été accepté. Il est présentement utilisé pour tous les nouveaux projets.

Par la suite, il y a eu un changement dans les priorités de la division de R&D qui a eu comme effet de suspendre l'implantation de RUP.

DISCUSSION ET INTERPRÉTATION DES RÉSULTATS

L'utilisation d'un processus de développement commercial comme RUP. comprend des avantages considérables. Il est plus avantageux d'avoir accès à une multitude d'informations sur le processus. de consulter les explications des activités et de consulter les modèles de documents de RUP que de repartir avec une feuille blanche. La formation reçue par Rational. sur la méthodologie d'implantation d'un processus dans une organisation m'a aidé beaucoup dans mon projet.

Malgré les avantages de RUP. son implantation dans la division R&D n'a pas été un succès. Nous avons identifié deux principales raisons pour cet échec : la direction de la division ne supportait pas l'effort d'un changement majeur dans l'approche du développement et le choix du plan d'implantation n'a pas fait ressortir la valeur de ce processus. Le choix de commencer l'implantation par l'enchaînement d'activités «gestion des exigences» est une erreur. L'impact du changement dans la définition des activités et l'implémentation d'un nouvel artefact n'est pas suffisamment visible pour justifier de continuer l'implantation de ce processus.

Nous aurions dû commencer par l'implantation de la phase d'inception et se concentrer sur l'enchaînement d'activité «gestion de projet». À ce moment. nous aurions introduit une meilleure planification de la mise en marché d'une nouvelle version des produits de Tecsys. Par contre. pour ce faire il nous fallait un engagement de la part de la direction et ceci est un élément important que nous n'avions pas. La direction était d'accord pour réviser les artefacts et mieux redéfinir les activités mais elle n'était pas d'accord à modifier de façon fondamentale la direction des projets.

Quelques mois après la suspension du projet pour la révision et l'amélioration des processus. le vice-président de la division a été remplacé et Tecsys a engagé un vice-président senior ayant un bagage dans la gestion de projet et dans l'assurance qualité. Depuis son arrivée. sans nécessairement suivre RUP. toutes les activités et les concepts

de la phase d'inception de RUP sont imposés et suivis pour la majorité de nos projets. Sans pour autant affirmer que RUP est le cycle de développement qui sera implémenté dans la division de R&D, il est certain que l'amélioration des processus en vue d'amener cette division à un niveau de qualité supérieure est désormais une priorité pour la nouvelle direction.

CONCLUSION

Durant le projet, j'avais le rôle d'architecte des processus. Ce rôle n'avait pas suffisamment d'importance au niveau de la charte organisationnelle. Ce rôle, à mon avis, doit être bien positionné dans l'entreprise afin d'avoir une certaine autorité. Ce rôle doit également être appuyé par la direction.

Suite à mon expérience je suis venu à la conclusion que le choix des processus à implanter n'est pas l'élément déterminatif dans le succès du changement. Que ce soit RUP ou autres, le soutien de la haute direction est le facteur de succès le plus important. Les changements de culture ne peuvent pas être imposés par les employés mais par la haute direction.

BIBLIOGRAPHIE

Jacobson. Ivar. Booch.Grady. Runbaugh. James (1999). The Unified Software Development Process. Addison Wesley.

Royce. Walker (1998). The software project management. Addison Wesley.

ANNEXE 1

Cycle de développement en cascade

TECSYS

Research & Development

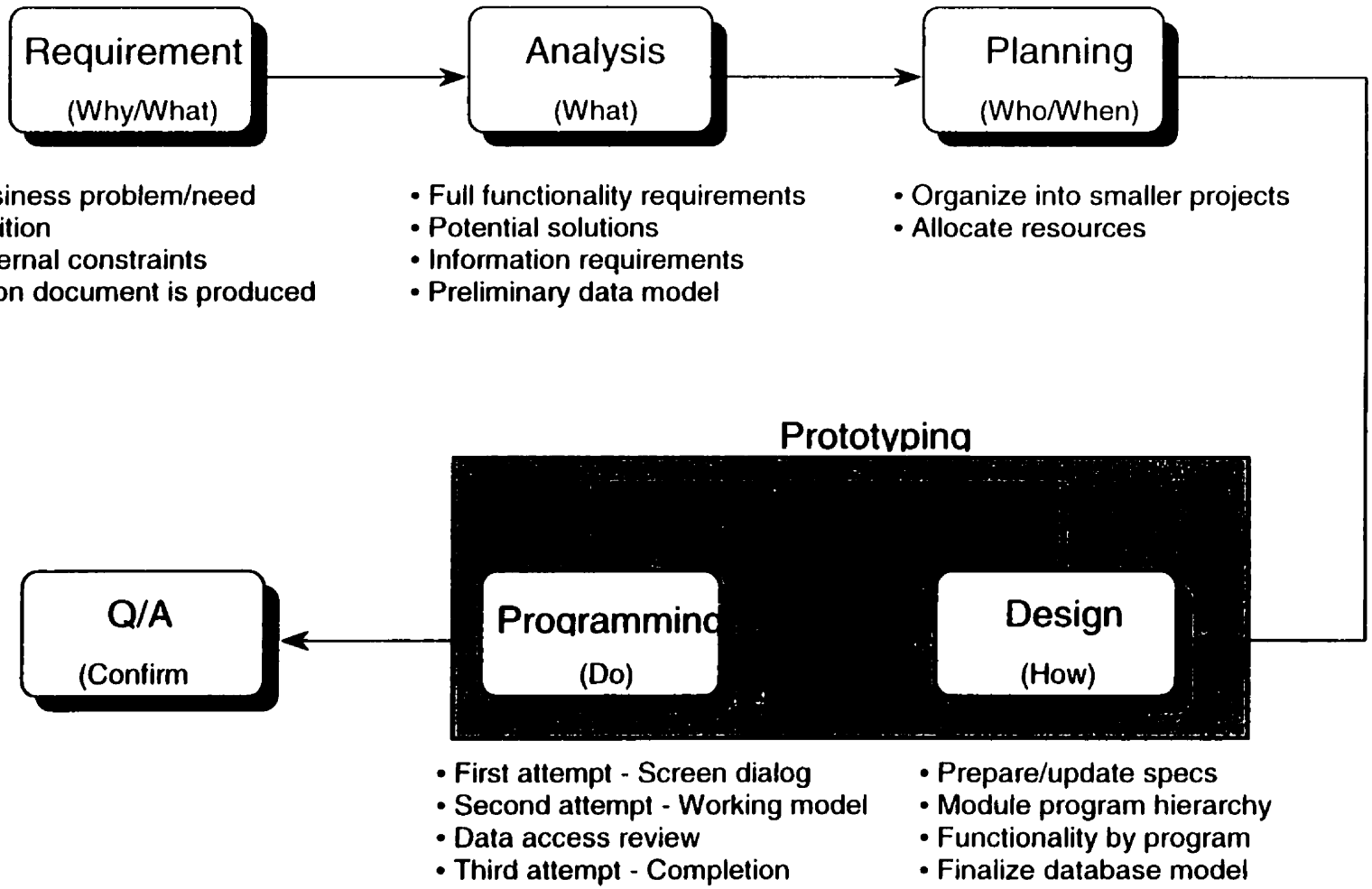
Procedure

R&D Development Process

August 28,2000

Table of Content

1. 6 Step Process - Waterfall



ANNEXE 2

Rapport - Équipe de direction de produit

TECSYS

RESEARCH & DEVELOPMENT

Business Process Design

Current State Analysis
Product Conception and Analysis

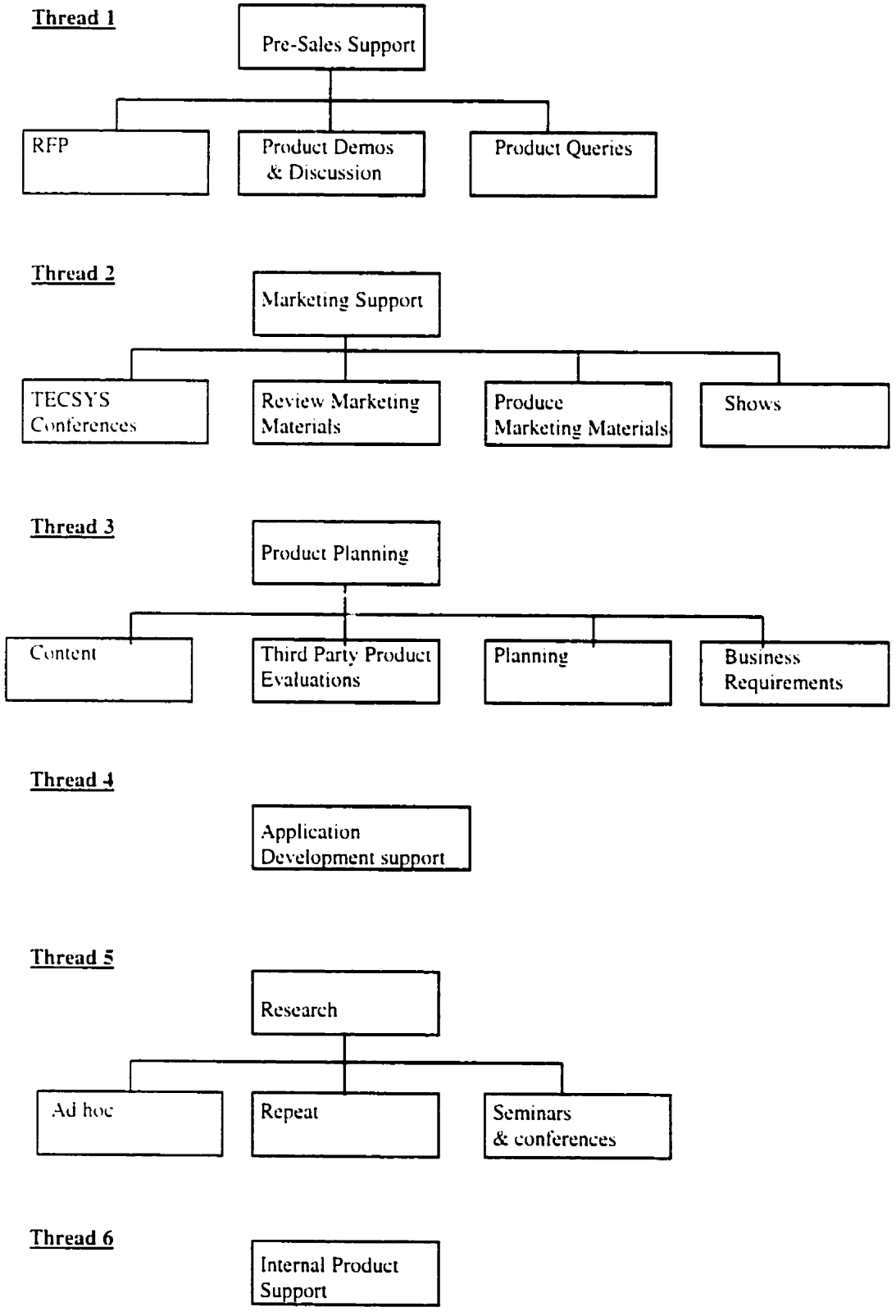
Final Version

August 28th 1999

TECSYS INC.
1840 Trans Canada
Dorval, Quebec
H9P 1H7
Tel.: (514) 333-0000
Fax: (514) 333-0109

Table of Contents

Process Hierarchy



Process Thread	Trigger	End
Pre-Sales Support		
RFP	Request from sales	Send completed RFP to sales
Product Demo & Discussion	Request from sales	Send write-up to sales
Onsite-Demo		
Offsite-Demo		
Remote-Demo		
Product Queries	Question from sales/tele-marketing	Answer query
Marketing Support		
TECSYS Conferences	Request from Marketing	Return to office
Shows	Request from Marketing	Return to office
Review Marketing Materials	Request from Marketing	Send comments/corrections
Produce Marketing Materials	Request from Marketing	Send document
Product Planning		
Content	Meeting scheduled	Release content, critical path
Third Party Product Evaluations	Request from Marketing	Decision made
Planning	Prod Content Document	Plan finalized
Business Requirements	Requirement Requested	Business reqmt. Document
Application development support	Request made/document received	Answer/review/comments given
Research		
Ad hoc/maintain notes	Idea triggered from various sources	Notes compiled/updated
Seminars and courses	Announcement recvd	Return to office/make notes
Internal Product Support	Request made	Answer given

Pre-Sales Support

RFP

1.Request comes from sales
2.Research questions
3.Answer questions
4.Send completed RFP back to sales

Product Demonstration & Discussion

Onsite-Demo

1.Request from sales
2.Get background description of prospect
3.Establish meeting room, equipment to be used
4.Prepare, get familiar with database
5.Attend meeting, do demo, discussions, make notes
6.Send write-up to sales

Offsite-Demo

1.Request from sales
2.Get background description of prospect
3.Get equipment to be used
4.Prepare, get familiar with database
5.Travel to prospect
6.Attend meeting, do demo, discussions, make notes
7.Return to office
8.Send write-up to sales

Remote-Demo

1.Request from sales
2.Get background description of prospect
3.Establish room, equipment to be used
4.Prepare, get familiar with database
5.Do remote demo, discussions, make notes
6.Send write-up to sales

Product Queries

1.Question from sales/telemarketing
2.Optionally research answer
3.Answer query

Marketing Support

TECSYS Conferences

1.Request from Marketing
2.Prepare material/presentation
3.Travel to conference
4.Give presentation
5.Make notes about questions, comments
6.Return to office

Shows

1.Request from Marketing
2.Travel to show
3.Make presentations, discussions with prospects
4.Make notes on serious prospects
5.Return to office

Review Marketing Materials

1.Request from Marketing
2.Read documents, make notes
3.Send comments/corrections

Produce Marketing Materials

1.Request from Marketing
2.Do research, prepare notes
3.Produce document
4.Send document

Product Planning

Content

Schedule content planning Meeting.

1. Including Peter Brereton, R&D, Sales, Professional Services and Marketing.
2. Review existing custom enhancements for generic retro-fit, customer commitments, short term product features.
3. White board all points and priorities by area.
4. Attempt to establish budget.
5. Select critical path for release. Identify minimum release content.

Planning

After Content meeting is complete a meeting is scheduled with all core R&D teams represented.

1. Using Content meeting priorities reality check requirements against resources.
2. Identify additional product requirements (technical or functional) that may have been omitted from Content Planning meeting.
3. Each core team performs production planning to verify content feasibility.
4. A Product Planning document is produced by all core team members providing release content write-ups.
5. Monthly meeting to identify current development plans and adjustments. Return to point (2)

Business Requirements

1. Based on a Content decision, a request is made for a requirement document.
2.Research and notes are prepared.
   What does the competition offer
   What does sales require
   What does our customer require
   etc. ...
3.Actual requirement document is produced using a requirements template.
4.Kick-off occurs as part of the development cycle when the analysis is assigned.

Third Party Product Evaluations

Request from President or Marketing.
Identify requirements
Research product documentation
Attend product demonstrations
Possible TECSYS meeting to discuss impact
Possible technical evaluation of third party product.
Negotiation on pricing for product occurs with Marketing.
Decision made

Application development support

Request made/document received
Research as required.
Answer/review/comments given

Research

Ad hoc/ Maintain research notes

Idea triggered from email (announcement, newsletter), other sources
Research subject (internet, etc.)
Notes compiled/updated

Seminars and courses

Announcement received
Research as required to determine whether investment is worthwhile/timely.
Approval obtained
Travel to location
Return to office, make/update notes

Internal Product Support

Request made (usually via email)
Research as required.
Answer given (may require conference call between PS, PM and customer).

Business Rules

Business Rules define the rules that govern the process.

Business Rules ...

#	Business Rule	Enforced	Effective
1	Content for RFP's must include both Generic and Custom Catalog functionality	N	?
2	Conferences and Demo's must have proper dress code applied	Y	Y
3	Analysis documents are signed off by product management after evaluation of completeness and applicability	Y	Y
4	Peter Brereton approves product content	Y	N
5	Projects must be monitored for scope creep	Y	N
6	Projects are evaluated against the budget and functionality cut accordingly if possible	N	N
7	Development and look-and-feel standards must be respected	N	N
8	Detail Spec documents are signed off by product management after evaluation of usability and look-and-feel	Y	N
9	Designated PS rep must provide feedback on proposed solution	N	N
10			
11			
12			
13			
14			
15			
16			
17			
18			

Performance Factors

Performance Factors define how the process is to be measured in specific terms, and captures whether the measurement is actually tracked in today's operations, along with it's value.

Performance Measures ...

#	Category	Performance Description	Measurement	Track	Value
1	Quality	Thoroughness of Requirements in terms of ensuring all components have been completed	Compare final Requirements spec to template.	N	M
2	Efficiency/Timeliness	Planning/Scheduling accuracy - adhering to the R&D schedule	Completion Date vs Planned Completion Date	Y	M

Tools

The purpose of the Tool table is to define the various tools and or systems that are required to facilitate the process.

Tools ...

#	Tool	Purpose / Use	Comments	Value
1	Analysis Template - Word	Ensure consistency and completeness	Low value due to format and lack of details	L
2	Flowcharter	Produce business flow, process flow, etc.	OK for flowcharts but not suited for business process diagrams	M
3	Internet	Research theories, business processes, competitive intelligence, third party potential, etc.	Requires much effort, patience - and both a fast machine and a high-speed connection	H
4	Powerpoint	Presentations for shows, prospects, conferences, internal training, etc.	Effective but time-consuming	H
5	Data Model (S-designer)	Reference tool for current data model.	Low value due to format limitations	L

Constraints

The constraint table is used to identify the constraints that have been imposed on the process.

Constraints …

#	Constraint	Impact
1	Effort permitted is often limited due to predefined time/cost constraints - jeopardizing best possible solution	M
2	Limited number of resources compared to demands and mandate	H
3	Timeline/scheduling of R&D projects, prospect visits, shows, and conferences - many conflicts	H

Constraints

The constraint table is used to identify the constraints that have been imposed on the process.

Constraints …

#	Constraint	Impact
1	Effort permitted is often limited due to predefined time/cost constraints - jeopardizing best possible solution	M
2	Limited number of resources compared to demands and mandate	H
3	Timeline/scheduling of R&D projects, prospect visits, shows, and conferences - many conflicts	H

Positive Aspects

The following table is used to capture all the positive aspects of the current process, in terms of anything that is working well and should potentially remain unchanged.

What works well ...

#	Positive Aspect	Comments	Value
1	Formal Requirements	Preparing formal Requirements is an excellent practice	H
1	Research	Provides background knowledge for business requirements, etc.	H
3	Knowledge Transfer	Allows for training and advancement of analysts and programmers	H
5	Presales, shows, seminars	Provides feedback from prospects, industry trends, evaluation of competition, third party research	H

Negative Aspects

The following table is used to capture all the negative aspects of the current process, in terms of anything that is not working well and should reviewed for potential changes. This table is not used to capture ideas to improve the problem, rather, it is used to simply identify the issue.

What doesn't work well ...

#	Negative Aspect	Comments	Impact
1	Tools	No tools for knowledge repository - no cumulative benefit from each project	H
2	Resources	We often assign resources based on availability as opposed to best fit - the analyst may not have appropriate experience or knowledge in subject matter	H
3	Resources	We often assign resources who have not been involved in project from beginning such as a Designer who was not involved in the analysis	M
4	Resources	Lack of sufficient resources in other departments (e.g. marketing, pre-sales)	H
5	Frustration level due to incomplete solutions	Occasionally Analyst will be frustrated because we are not permitted to flush out the best possible solution - we are mandated to "keep things simple"	L
6	Lack of knowledge and/or training in other departments	Leads to 'bombardment' of requests for information, assistance, etc.	H
7	Ever-changing schedules and priorities	Puts stress on time management and reduces quality/timeliness of deliverables	H
8	Lack of accessibility and/or completeness of information	Leads to 'bombardment' of requests for information, assistance, etc. from other departments. Also impacts efficiency in new projects.	H
9	No executive commitment to a clear corporate direction	Affects product planning and feature contents (schedules and priorities), increases frustration and confusion.	H

Critical Success Factors

The following table represents those factors which are deemed to be absolutely critical to the success of the process and deliverables alike. In other words, if these factors are not applied, we are sure to fail.

Critical Success Factors ...

#	Critical Success Factor	Impact of NOT Implementing
1	Long term commitment to a clear and concise corporate direction must exist	Wasted time in focusing on areas that get changed or dropped because of no commitment
2	Product planning sessions and release content 'announcements'	Sales will continue making generic commitments beyond what's available or for sooner than planned.
3	Cross-product synergy (e.g. WMS-DMS, FMS-Elite, etc.)	Conflicts in priorities and no coordination of features
5	Alignment of Corporate expectations with current resources (sales, marketing, executives should not compare us with competitors having 10 times or more resources)	Constant dissatisfaction with product capabilities and development progress

ANNEXE C

RAPPORT

EQUIPE DE DÉVELOPPEMENT

ANNEXE 3

Rapport - Équipe de développement

TECSYS

RESEARCH & DEVELOPMENT

Business Process Design

Current State Analysis
Application Development

Final Copy

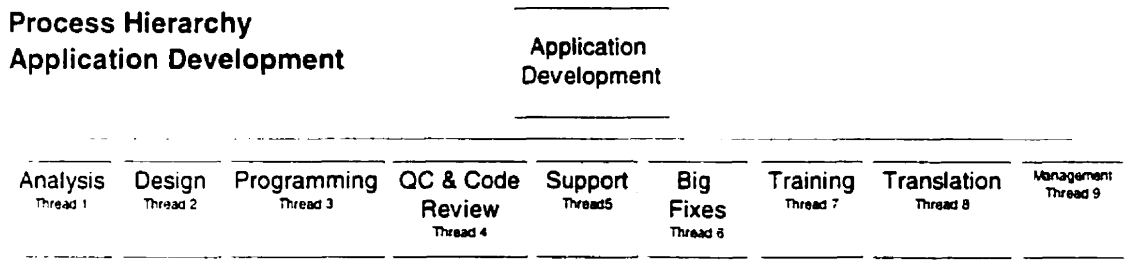
August 28th 1999

TECSYS INC.
1840 Trans Canada
Dorval, Quebec
H9P 1H7
Tel.: (514) 333-0000
Fax: (514) 333-0109

Table of Contents

1. PROCESS HIERARCHY

1.1. HIGH LEVEL PROCESS HIERARCHY



1.2. PROCESS HIERARCHY - TEXTUAL

1.2.1. Thread 1 - Analysis

EBP 1. Trigger → Receive project assignment from manager.
EBP 2. Open job number to post analysis time.
EBP 3. Review written business requirements if available.
EBP 4. Establish sources of information and knowledge.
EBP 5. Kick-off meeting.
EBP 6. Identify approach & methodology for analysis.
EBP 7. Interview knowledge sources. *
EBP 8. Provide manager with ballpark estimate for initial iteration of analysis.
EBP 9. Plan project. Refer to management thread for further information.
EBP 10. Flesh out business requirements as new information presents itself.
EBP 11. Prepare functional requirements.
EBP 12. Obtain Product Managers verbal agreement on functional requirements.
EBP 13. Ponder alternatives.
EBP 14. Test functionality of existing modules. *
EBP 15. Determine feasibility of alternatives.
EBP 16. Prepare proposed solution(s).
EBP 17. Question knowledge sources (iterative). *
EBP 18. Revise analysis and document accepted solution(s). *
EBP 19. Prepare entity relationships. *
EBP 20. Prepare system process flows. *
EBP 21. Prepare preliminary data model.
EBP 22. Prepare other forms of supporting documentation. (eg. mapping guides). *
EBP 23. Evaluate impact of subject matter on EliteDMS.
EBP 24. Evaluate impact of subject matter on EliteWMS. *
EBP 25. Evaluate impact of subject matter on EliteQ. *
EBP 26. Evaluate impact of subject matter on EliteView. *
EBP 27. Review document and reorganize information in logical sequence.
EBP 28. Examine whether documentation covers off on all aspects of the requirements.
EBP 29. Review document with another analyst. Review by peers. *
EBP 30. Revise document as a result of peer feedback.
EBP 31. Review document with DBA to review solution.
EBP 32. Revise document as a result of DBA feedback.
EBP 33. Review document with product manager.
EBP 34. Revise document as a result of P.M. feedback.
EBP 35. Send document to product management for final approval.
EBP 36. Prepare ballpark estimate for next phase of development.
EBP 37. Review estimates and costs of next phase against budget.
EBP 38. Revisit portions of analysis to accommodate budget related issues. *
EBP 39. Issue status reports (on request by team and/or R&D manager).
EBP 40. Log time spent during entire process in the time-job system.
EBP 41. Trigger ← Receive final approval of analysis document - Analysis complete.

* Conditional (requires a decision box).

1.2.2. Thread 2 - Design

EBP 1. Trigger → Analysis (or portion of) is complete.
EBP 2. Open job number to post design time.
EBP 3. Review analysis. (May be abbreviated/verbal depending on the type of project.)
EBP 4. Identify resources.
EBP 5. Subdivide tasks. *
EBP 6. Attend design kick-off meeting to review requirements and analysis. (Review business requirements, solution, data model, etc.)
EBP 7. Identify approach & methodology for design (high level vs. detailed).
EBP 8. Identify programs affected (if not provided in the analysis) before the quote. *
EBP 9. Review existing ballpark estimate for design with Analyst.
EBP 10. Review existing ballpark estimate for design with manager.
EBP 11. Ponder/investigate design issues.
EBP 12. Create a design quote.
EBP 13. Review quote with team manager.
EBP 14. Revise quote as a result of feedback from team manager.
EBP 15. Team manager approves quote.
EBP 16. Manager submits Quote.
EBP 17. Program design for EliteDMS, EliteWMS, EliteQ, EliteView: Steps 18 thru 23 loop for each program affected.
EBP 18. Design screen layout /user interface issues. *
EBP 19. Identify validation, defaults, messages and help for each field / screen. *
EBP 20. Define the processing. (potentially some pseudo code) *
EBP 21. Define database interactions. (inserts, deletes, updates, etc.) *
EBP 22. Prepare/update program general help. *
EBP 23. Design the report layout / modifications. *
EBP 24. Update project data model (final version). *
EBP 25. Verify that the design did not miss any affected programs or help.
EBP 26. Review document and reorganize information in logical sequence.
EBP 27. Revise other forms of supporting documentation (eg. mapping guides). *
EBP 28. Examine whether documentation covers off on all aspects of the analysis.
EBP 29. Review the design with analyst.
EBP 30. Revise document as a result of analyst feedback (if required). *
EBP 31. Review document with product manager.
EBP 32. Revise document as a result of P.M. feedback. *
EBP 33. Review document with DBA
EBP 34. Revise document as a result of DBA feedback. *
EBP 35. Update analysis document if required. *
EBP 36. Create initial Test Plan – list of features requiring testing and testing considerations.
EBP 37. Send design document to product management for final approval.
EBP 38. Prepare estimate for next phase of development.
EBP 39. Review estimates and costs of next phase against budget. *

EBP 40. Revisit portions of design to accommodate budget related issues. *
EBP 41. Log time spent during entire process in the time-job system.
EBP 42. Issue status reports (on request by team and/or R&D manager).
EBP 43. Trigger ← Receive final approval of design document - Design complete.

* Conditional (requires a decision box).

1.2.3. Thread 3 - Programming

EBP - 1. Trigger → Receive project assignment from manager.
EBP - 2. Open job number to post design time.
EBP - 3. Assign resources to programming tasks.
EBP - 4. Review analysis and/or design document(s).
EBP - 5. Attend programming kick-off meeting.
EBP - 6. Prepare programming estimates (quote).
EBP - 7. Review programming estimates (quote) with project designer/programmer.
EBP - 8. Revise estimate based on feedback from project designer/programmer.
EBP - 9. Submit programming estimates (quote) for approval by team manager.
EBP - 10.Review programming estimates (quote) with team manager.
EBP - 11.Revise estimate based on feedback from team manager. *
EBP - 12.Submit programming estimates (quote) for approval by R&D manager.
EBP - 13. Approve programming estimates (quote approval by R&D manager).
EBP - 14. Apply database changes to development environment. *
EBP - 15. Copy (check-out) and/or create option or library.
EBP - 16. Define new programs/libraries in Elite's menu structure. (EliteDMS only)
EBP - 17. Create/modify screen forms (based on design document if screen captures are included).
EBP - 18. Write 4GL code (using established standards and existing software resources/utilities).
EBP - 19. Revisit issues with designer (iterative). *
EBP - 20. Write online help (point form, often excerpts of design document).
EBP - 21. Submit request(s) for new messages to Translation Dept. (EliteDMS only)
EBP - 22. Modify la2 forms in order that they match the English.
EBP - 23. Modify la2 help to include new section numbers.
EBP - 24. Compile code, forms and help.
EBP - 25. Perform Programmer's QC (using design document and standard Development Checklist*).
EBP - 26. Update 4GL code with message number(s) identified by Translation Dept. and re-test.
EBP - 27. Revise and enhance test plan.
EBP - 28. Log calls in the TimeJob Reporting System for any bugs not related to current modification.
EBP - 29. Discuss (with project leader) and possibly fix critical bugs found that are not related to current modification.
EBP - 30. Log/track programming time using TimeJob Reporting System.
EBP - 31. Issue status reports (on request by team and/or R&D manager).

EBP - 32. Make options/libraries available for Q/C by project leader. (different processes for EliteDMS & EliteWMS)
EBP - 33. Trigger ← Programming Complete.
EliteView ?? *
EliteQ ?? *

* Conditional (requires a decision box).

1.2.4. Thread 4 - Quality (Code Review & QC)

EBP 1. Trigger → Receive project assignment from manager.
EBP 2. Review analysis and design document(s).
EBP 3. Attend programming or QC kick-off meeting. *
EBP 4. Perform code review using combination of Development Checklist, "SQL explain" and design document.
EBP 4. 1.Compare code against Development checklist/coding standards.
EBP 4. 2.Verify code structure and readability
EBP 4. 3.Verify transaction scopes
EBP 4. 4.Verify logic in terms of performance and functionality.
EBP 5. Document findings using standardized QC Form.
EBP 6. Present findings to programmer, notify project leader.
EBP 7. Programmer makes fixes based on CR's findings (QC Form).
EBP 8. Review code changes resulting from code review fixes.
EBP 9. Inform project leader that code review is complete.
EBP 10. Perform unit tests using combination of Development Checklist, test plan and design document.
EBP 11. Revise and enhance test plan.
EBP 12. Document findings of unit tests using standardized QC Form.
EBP 13. Present findings to programmer, notify project leader.
EBP 14. Programmer makes fixes based on QCer's findings (QC Form).
EBP 15. Test fixes resulting from bugs logged during Unit Testing.
EBP 16. Code reviewer rechecks code. *
EBP 17. Perform integration tests comparing results against analysis and design documents / resources.
EBP 18. Revise and enhance test plan.
EBP 19. Document findings of integration tests using standardized QC Form.
EBP 20. Present findings to programmer, notify project leader.
EBP 21. Programmer makes fixes based on QCer's findings (QC Form).
EBP 22. Test fixes resulting from bugs logged during Integration Testing.
EBP 23. Log/track QC time against programming job code using TimeJob Reporting System.
EBP 24. Issue status reports (on request by team and/or R&D manager).
EBP 25. Advise programmer(s) that QC has been completed.
EBP 26. Programmer releases code.
EBP 27. Trigger ← CR & Q/C Complete.
EliteView ?? *

EliteQ ?? *

* Conditional (requires a decision box).

1.2.5. Thread 4 - Analyst QC

EBP 1. Trigger → Receive advisory that programming is complete.
EBP 2. Review project plan and determine status of all sub-tasks.
EBP 3. Determine plan of attack.
EBP 4. Update test plan. *
EBP 5. Review project as a whole (run programs, review help, etc.) to ensure it generally meets functional requirements.
EBP 6. Verbally discuss variations with programmer involved.
EBP 7. Document situation using QC Form. *
EBP 8. Trigger ← Inform project leader that process is complete.

* Conditional (requires a decision box).

1.2.6. Thread 5 - Support to Professional Services Group

EBP 1. Trigger → Receive request for help (Telephone, email, third party).
EBP 2. Interview PS rep to further determine/understand the problem/question/situation.
EBP 3. Interview the client to further determine/understand the problem/question/situation.
EBP 4. Determine if bug or misuse/misunderstanding.
EBP 5. Provide requirements, analysis and design documents.
EBP 6. Explain solution/limitations or provide other supporting documents.
EBP 7. Log call in TimeJob System (if issue relates to bug/enhancement). *
EBP 8. Log/track time against proper job code using TimeJob Reporting System.
EBP 9. Trigger ← PS is informed of functionality/limitations.

1.2.7. Thread 5 - Support to Custom Enhancements Group

EBP 1. Trigger → Request for help (via telephone, email).
EBP 2. Discuss issue (background information) with C/E resource.
EBP 3. Perform quick search through code/documentation. *
EBP 4. Answer questions, give opinions based on experience.
EBP 5. Consult with other resource(s). *
EBP 6. Explain solution/limitations or provide other supporting documents.
EBP 7. Suggest sources for further reference (R&D resources, Product Direction, documentation, etc.).
EBP 8. Log call in TimeJob System (if issue relates to bug/enhancement). *
EBP 9. Log/track time against proper job code using TimeJob Reporting System.
EBP 10. Trigger ← C/E is informed of functionality/limitations.

* Conditional (requires a decision box).

1.2.8. Thread 5 - Support to Support Group

EBP 1. Trigger → Support requires assistance.
EBP 2. R&D checks the problem.
EBP 3. Determine whether issue is a bug or related to training.
EBP 4. Individual finds someone to give assistance (if not qualified). *
EBP 5. Explain solution/limitations or provide other supporting documents.
EBP 6. Suggest sources for further reference (R&D resources, Product Direction, documentation, etc.).
EBP 7. Support/R&D enters a call(s). **
EBP 8. Immediately fix high priority problems during transitional period(s). *
EBP 9. Release Management reviews calls and forwards "special" calls to R&D. **
EBP 10. Log/track time against proper job code using TimeJob Reporting System.
EBP 11. Trigger ← C/E is informed of functionality/limitations.

Note: If not high priority then will schedule for later fix for the final cut.

* Conditional (requires a decision box).
** Special could mean: beta related calls, calls that require a large amount of time/effort/expertise, re-design, re-work, etc.

1.2.9. Thread 5 - Support to Pre-sales/Sales

EBP 1. Trigger → Pre-sales and/or Sales contacts R&D for support
EBP 2. Individual finds someone to give assistance (if not qualified). *
EBP 3. Demonstrate W.I.P. features.
EBP 4. Demonstrate knowledge in specific areas to prospects. (Business/technical).
EBP 5. Provide estimates (ballparks) for potential modifications/new features.
EBP 6. Explain solution/limitations or provide other supporting documents.
EBP 7. Create a prototype. *
- EBP 7. 1.Anl/Des step - (non documented specification research)
- EBP 7. 2.Review design with pre-sales (verbally)
- EBP 7. 3.Create prototype development environment
- EBP 7. 4.Create/modify programs as required
- EBP 7. 5.Create/modify database tables
- EBP 7. 6.Create/modify help
- EBP 7. 7.Demo to pre-sales
- EBP 7. 8.Install on pre-sales demo machine
- EBP 7. 9.Perform 3-D effect conversion
- EBP 7. 10.Test on pre-sales demo machine

EBP 8. Suggest sources for further reference (R&D resources, Product Direction, documentation, etc.).
EBP 9. Log/track time against proper job code using TimeJob Reporting System.
EBP 10. Trigger ← Sales is informed of functionality/limitations.

* Conditional (requires a decision box).

1.2.10. Thread 6 - Bug Fixing

From QA (for an open project)

EBP 1. Trigger → Promote project to QA.
EBP 2. Inform QA that the project is ready.
EBP 3. QA goes through the analysis & design documents. *
EBP 4. QA Kick-off with assigned R&D resource.
EBP 5. QA executes QA process (See Quality Assurance Section)
EBP 6. QA finds problems and enters a new ("NE") call in time job / type "QP".
EBP 7. QA contacts R&D (project leader to make them aware).
EBP 8. Project leader looks at bug calls - prints call.
EBP 9. Evaluate call (bug versus non-bug/enhancement/training issue/limitation).
EBP 10. Project leader dispatches bug fixes to resources.
EBP 11. Programmer set the call status to "IP". *
EBP 12. Programmer fixes bug and sets call status to "QC".
EBP 13. Project leader QC's bug fixes.
EBP 14. Project leader then advises programmer to release & promote to QA.
EBP 15. Project leader sets the status to QA in time job.
EBP 16. QA then picks up all "QA" status calls.
EBP 17. If QA rejects a bug fix ➔ set the status to "RJ".
EBP 18. QA advises R&D of status change ("RJ") & call numbers to project leader for the job.
EBP 19. When QA is satisfied with the project quality, then QA advises R&D Manager.
EBP 20. Time recorded in Time/job against job number - labor code "5Q".
EBP 21. Trigger ● R&D Manager closes job.

For current release

EBP 1. Trigger → Print out bug list (a report for "QP" type T/J entries).
EBP 2. Team Manager reviews calls.
EBP 3. Managers define available resources to do bug fixes on current release.
EBP 4. Team manager E-mails the bug numbers to the available resources.
EBP 5. Programmer set the call status to "IP". *
EBP 6. Programmer fixes bug and sets call status to "QC".
EBP 7. Team Manager / Project Leader Assigns QC resource & code review resource.
EBP 8. R&D resource QC's bug fixes.
EBP 9. QC'er sets call status to "QA" and advises programmer.
EBP 10. Programmer releases and promotes programs to QA.
EBP 11. QA tests fixes.
EBP 12. QA closes or rejects the call.
EBP 13. If QA rejects a bug fix ➔ set the status to RJ.
EBP 14. QA advises R&D of status change & call numbers to project leader.
EBP 15. Trigger ← QA closes call.

Note: R&D still fixes bugs on current release projects, even if they are closed - except bill 900/999 with labor code of 5C. After release is cut the call ends up in release management.

* Conditional (requires a decision box).

1.2.11. Thread 7 - Training

EBP 1. Trigger → Manager communicates training requirements.
EBP 2. Manager assigns training resource.
EBP 3. Identify attendees and their knowledge levels.
EBP 4. Confirm students and expectations with student supervisor/client contact. *
EBP 5. Prepare training material including examples and exercises (labs). *
EBP 6. Review materials for target audience.
EBP 7. Prepare agenda and send to client/students. *
EBP 8. Prepare physical resources (book room, teaching aids, hardware, software, environments, and travel arrangements). *
EBP 9. Teach course.
    EBP 9. 1.Present material.
    EBP 9. 2.Review examples.
    EBP 9. 3.Give practice sets.
    EBP 9. 4.Evaluate at end of course. *
EBP 10. Evaluate students in written email to their manager(s). *
EBP 11. Log time spent during entire process in the time-job system.
EBP 12. Trigger ← Training complete.

* Conditional (requires a decision box).

1.2.12. Thread 8 - Translation

Planning of English - French New/Maintenance Release

EBP 1. Trigger → New feature or maintenance release is cut.
EBP 2. Software Engineering creates translation environment. (Directory structure & changes to scripts, etc.).
EBP 3. Plan translation tasks using the compfh script as well as taking into account the release schedule (upcoming projects).
EBP 4. Review "D9"'s to determine is they can be fixed along with other features.
EBP 5. Generate estimates using best guess and past experience. (Tool: time_est script).
EBP 6. Team manager adds the new tasks in the project plan.
EBP 7. Team manager assigns resources.
EBP 8. Team manager distributes plan to team members. (individually & globally).
EBP 9. Translators generate the CodeWrite diffs → placed in /DOC/6-1/diffs
EBP 10. Trigger ← planning of translation complete.

DTP develops for Standardization & Formatting purposes *

EBP 1. Trigger → Receive planning schedule.

EBP 2. Copy (check-out) option or library (take ownership of program).
EBP 3. Format & standardization:
    EBP 3. 1.New Programs:
        EBP 3. 1.1.Give new programs a DOC-ID code (Identify to which module it belongs).
        EBP 3. 1.2. Convert help into .txt format for Word. (Tool: convert_help).
        EBP 3. 1.3.Re-format .txt file using a standard Word template.
    EBP 3. 2.Existing Programs:
        EBP 3. 2.1.Create back-up.
        EBP 3. 2.2.Open the CodeWrite diff file
        EBP 3. 2.3.Incorporate changes into existing Word document.
EBP 4. Standardize terminology, formats, fonts, etc.
EBP 5. Export the file & compile programs.
EBP 6. Run program??
EBP 7. Update control spread sheet (control doc??) with date & give ownership (giveown) to translator.
EBP 8. Produce diff of standardized work.
EBP 9. Trigger ← Advise translator that the program is ready.

Programs are translated by translators

EBP 1. Trigger → Receive notice from DTP that program is ready.
EBP 2. DTP gives ownership of programs to translators.
EBP 3. In the case of bugs / maintenance release - translator takes ownership in all environments (current release & maintenance release env.)
EBP 4. Translator goes into Word and incorporates standardized diff into the second language.
EBP 5. Fix translations bugs ("D9") as per team managers plan.
EBP 6. Translate diction table, 4gl ring menus, help (Word document), and forms.
EBP 7. Translator updates the control document with the translation date.
EBP 8. Ownership is passed back to DTP for format & functional QC.
EBP 9. Create final diff and save in the holding directory.
EBP 10. Update the control spreadsheet with date and indicate the existence of diff file (for Ref. Guides).
EBP 11. Release the program(s) (Tool: transrel script)
EBP 12. Promote program to Smaster.
EBP 13. Trigger ← Translation cycle is complete.

Production of Reference Guides

EBP 1. Trigger → Reference guide required according to plan.
EBP 2. Create new directory structure in framemaker.
EBP 3. Copy the master reference guide template into framemaker.

EBP 4. Update lib integration sheet (see control.doc - 6.1.xls)
(indicates which libs have to be incorporated in which books).
EBP 5. Get status from Control document: i.e. check Holding column (diff or no diff)
EBP 6. Decide if better to import file from scratch or make changes to previous file based on the diff (if there is a diff).
EBP 7. Import new or extensively changed help file from the directory structure on L: drive (to re-format from scratch).
EBP 8. Get previous version of framemaker file if incorporating diff only.
EBP 9. Format file as per guidelines.
EBP 10. Port all options into framemaker that were not touched - this includes all libraries (go to previous release framemaker directory structure).
EBP 11. Update control document as each option/library imported into framemaker.
EBP 12. Ensure no changes made since each option imported into ref. Guides (date of holding dir. Diff < import date).
EBP 13. Create a book list (file) for every module - framemaker file that contains a list of all files that make a book i.e. TOC, introduction, index, customer feedback sheet, all options. These are all linked together to make the book.
EBP 14. Update Preface, Front matter, Application menus are updated with new information or changes (from last version).
EBP 15. Integrate the libs in the appropriate book(s).
EBP 16. Integrate the new report samples into framemaker (reports & batches).
EBP 17. Generate the TOC and the index files.
EBP 18. QC of index & TOC.
EBP 19. Print a hard copy of the book (aprox. 20 books).
EBP 20. QC layout of each book.
EBP 21. Edit the book for changes to layout.
EBP 22. Regenerate the TOC & index (if necessary).
EBP 23. Print the final copy and final verification.
EBP 24. Create the .pdf files (portable document format) for each book (test links).
EBP 25. Get confirmation that .pdf file complete for report digest.
EBP 26. Create the READ.ME which is a notice of completion of the books which contains
a list of the books and report digest. User can click on any title to open book.
EBP 27. Post the Reference Guides & READ.ME to the Intranet (MIS & R&D Administrative Assistant).
EBP 28. Verify that all is accessible.
EBP 29. Trigger ← Email to the company that reference guides are now available .

Report Digest

EBP 1. Trigger → Translation gives a list of all reports to generate samples for report digest to R&D.
EBP 2. R&D generates data needed to generate adequate sample reports.
EBP 3. R&D generates the report samples.
EBP 4. R&D QC's the report samples.
EBP 5. R&D notifies translation when the job is done.
EBP 6. Create new book (framemaker file) for the report digest.
EBP 7. Copy old book from previous release.
EBP 8. Update the existing reports and insert new report samples into new book.
EBP 9. Format the file as per guidelines.
EBP 10. Generate TOC (Table of contents).
EBP 11. Print the report digest.
EBP 12. QC & Edit the report digest.
EBP 13. Re-print the report digest.
EBP 14. Trigger ← Create the .pdf files for the intranet.

Note that a bug fix or minimal changes may be done through translators from a to z.

English - Spanish New/Omega Release

EBP 1. Software Engineering creates translation environment. (Directory structure & changes to scripts, etc.).
EBP 2. At the end of a release, team manager runs a script that compares the last version of Spanish version's English against the new English version of latest release.
EBP 3. Team manager puts the result of the report in the plan.
EBP 4. Generate estimates using best guess and past experience. (Tool: time_est script).
EBP 5. Develop program. (Tool: SPDVLP script).
EBP 6. New Programs:
    EBP 6. 1.Convert help into .txt format for Word. (Tool: convert_help).
    EBP 6. 2.Re-format .txt file using a standard Word template.
    EBP 6. 3.Research Spanish terminology.
    EBP 6. 4.Translate.
EBP 7. Existing Programs:
    EBP 7. 1. Setup the word document structure. (Tool: osetup script).
    EBP 7. 2. Generate a CodeWrite diff to see what changes have been made. *
    EBP 7. 3. Incorporate the 4gl changes. (Tool: overlay script).
    EBP 7. 4. Review changes resulting from overlay script.
    EBP 7. 5. Translate libraries, options, diction table, forms and help (Word document).
    EBP 7. 6. Spanish translation errors ("D8") are completed when possible as they touch an option already in development.

EBP 8. Export the file & compile programs.
EBP 9. Update control spread sheet (control doc??) with date & give ownership (giveown) to DTP.
EBP 10. DTP compiles. QC's.
EBP 11. Release the program(s) (Tool: transrel script)
EBP 12. Promote program to Smaster.
EBP 13. Create final diff and save in the holding directory.
EBP 14. Update the control spreadsheet with date and indicate the existence of diff file (for Ref. Guides).
EBP 15. Trigger ← Translation cycle is complete.

Message Maintenance

EBP 1. Trigger → Request received from development team.
EBP 2. DTP creates messages.
EBP 3. DTP informs translation department that a message requires translation.
EBP 4. Translator translates messages & emails DTP when complete.
EBP 5. DTP QC's language 2 message text.
EBP 6. Trigger ← Message number is sent to person who made request.

* Conditional (requires a decision box).

1.2.13. Thread 9 - Management

- Related to a project
  - Ask for PO for Analysis job
  - Review design & programming quotes
  - Process design & programming quotes
  - Close jobs
  - Follow up on projects
  - Meeting Product management with schedule of what is coming
  - Customer contact / P.R.
  - Follow-up & assign tasks to documentation
  - Bug Fix follow-up & assignment of resources
  - Occasional QC
  - Occasional Bug Fix
  - Meeting with managers for project status updates to identify potential problem(s)
  - Resolve or escalate problem(s) to proper people
  - Communicate status to Tecsys management
  - Ensure that R&D. NTR. Release Management. Training. Translation. documentation are on the same page with respect to future releases
- Follow-up on customer commitments
  - Trigger → Email
  - Receive the customer contract
  - Read

    - Update spreadsheet (Customer contract commitments)
    - Ask questions to sales representative
    - Follow up that commitments made and carried through
    - Escalate issues to direction (when necessary)
    - Bring up issues in product planning sessions to decide if issues are to be addressed
- Product rollout
    - Meeting different managers to see what training is needed
    - When should the training should be done
    - Who should be trained
- Planning
    - Meeting to establish team status
    - Creation of the master plan
    - Continuous update of the master plan with day to day issues (ex: new tasks, project slippage etc.)
    - Manage resource conflict
    - Assign resources to the projects at the general level (# hours)
    - Meeting the group for direction changes
    - Follow-up with other depts on various issues (eg. WMS , PVCS ...)
    - Communicate deliveries to other departments
    - Help QA with the planning (temporary task)
    - Follow translation planning
    - Coordinate future releases with other dept.
    - Help coordinate training courses (help find resources)
    - Fill current release content report
- Administrative tasks
    - Billing
    - Time sheet verification
    - Verification of the month end reports
    - Budget (annual & project)
    - Review of Open Summary Task Report
    - Produce overtime report
    - Approve expense, overtime, training, contractual invoice and other forms.
    - Meeting (weekly with VP and all managers in R&D)
    - Miscellaneous (coordinate move, decorate etc.)
    - Re-organization of group → produce group hierarchy
    - Answer e-mail (all depts)
    - Payroll reports for part-time employees
    - H/W Problems
    - H/W and software requests
    - Purchase books or other materials
    - Improve processes

- Develop and ensure implementation of policies, procedures and quality standards
- Ensure interdepartmental collaboration

- Human Resources Aspect
  - Annual performance review
  - Annual salary reviews
  - Probation
  - Hiring
  - Interviews
  - Follow up on individual career
  - Manage employee personal problems
  - Follow-up on absenteeism, vacation and time-off
  - Startup time with new employees
  - Follow-up with H.R. re: employee issues/difficulties
  - Manage performance problems
  - Build replacements (back-up for DBA, R&D Manager, EliteQ, EliteView ...)
  - Train replacement (before vacation)
  -
- Training
  - Schedule training for the group on different aspects (ex: WMS, management training, software training etc.)
  - Coordinate with Yasmin's assistant course timetable
  - Coordinate with Yasmin's assistant course trainers
  - Coordinate with Yasmin's assistant course attendees
  - Coordinate with other Managers course trainer requirements *
  - Coordinate with the trainers and/or attendees the courses
  - Update the master plan to include all training aspect (prepararing, giving the course, attending the course etc.)
  - When required, coordinate with Yasmin's assistant the purchase of supporting documents.
  - Review results of course evaluations
  - Review results with trainer *

1.2.14. Thread 9 - Database Administrator

EBP 1. Trigger → R&D requests that DBA participate to Analysis kick-off.
EBP 2. Assist/develop in developing logical design of the database modifications.
EBP 3. Review analysis document (approach, architecture, preliminary model, performance requirements etc. ...).
EBP 4. Sign-off on analysis document.
EBP 5. Participates in Design Kick-off Meeting.
EBP 6. Develop/assist in detailed development of logical data model.

EBP 7. Create physical model and document required modifications.
EBP 8. Sign-off on detailed design.
EBP 9. DBA generates and applies scripts (and/or) programs to modify and/or incorporate changes to the database.
EBP 10. Updates physical model.
EBP 11. Support development team if required.
EBP 12. If needed develop and maintain new views (EliteView).
EBP 13. Nearing the end of a release DBA reviews Upgrade document to review all db changes.
EBP 14. Sign-off on upgrade scripts or programs (Does actual QC).
EBP 15. Reviews current physical data model changes with current database.
EBP 16. Updates data model with S-Designor 4.2
EBP 17. Trigger ← Release is cut.

2. BUSINESS RULES

Business Rules define the rules that govern the process.

Business Rules ...

#	Business Rule	Enforced	Effective
1	All documents should be updated when revisions are encountered.	N	H
2	Time must be tracked for all phases of a development project.	Y	H
3	Analysis and design documents must be formally approved/signed.	Y	H
4	All phases of development must be preceded by a kick-off meeting.	Y	H
5	Formal Business Requirement documents must exist.	N	H
6	Estimates must be approved prior to starting development phase.	N	H
7	Test plans must be created/update and used throughout the entire development process.	Y	M
8	Development environment must be set-up prior to start of programming/translation.	N	H
9	Quality must be measured and tracked throughout the development process (using existing forms).	Y	H
10	Group/individual must investigate issues before calling R&D for support.	N	H
11	Bugs must be properly logged and tracked.	Y	H
12	Project-related bugs logged against the current release must be prioritized and fixed.	N	H
13	All priority '1' bugs must be corrected prior to cutting a new release.	Y	H
14	All code changes must be code reviewed.	N	H
15	All code changes must be QC"ed.	Y	H
16	A development task is not complete until it is formally released by QA.	Y	H
17	The application must be standardized and translated to French prior to releasing the product.	Y	H
18	The application must be translated to Spanish in parallel to French translation.	N	H
19	Release content must be properly defined in advance.	N	H
20	Track customer contracts (for the purpose of reviewing commitments).	Y	H
21	Management must ensure that all corporate policies are followed.	Y	H
22	R&D must do its share as it relates to training (Internal and external)	Y	M

3. PERFORMANCE FACTORS

Performance Factors define how the process is to be measured in specific terms, and captures whether the measurement is actually tracked in today's operations, along with it's value.

Performance Measures ...

#	Category	Performance Description	Measurement	Track	Value
1	Quality	Thoroughness of Analysis Spec in terms of ensuring all components have been completed (ie: database design, database definitions, impact on elite, etc)	Compare final Analysis spec to template.	N	M
2	Quality	Thoroughness of Analysis in terms of proposing alternative solutions	Number of viable alternative solutions presented, including Pros/Cons & Recommendation	N	H
3	Efficiency/Timeliness	Scope Control	Number of scope increments including complexity	N	H
4	Efficiency/Timeliness	Review / Approval Period - Scheduling conflicts	Delay time between hand-offs and feedback	N	H
5	Cost	Estimation accuracy	Effort vs. Estimate in Hours	Y	M

4. TOOLS

The purpose of the Tool table is to define the various tools and or systems that are required to facilitate the process.

Tools ...

#	Tool	Purpose / Use	Comments	Value
1	Templates (Forms) - Word/Excel	Ensure consistency and completeness • Analysis • Design • QC Forms • Development Check List • Quotes • Test plans • Administrative (P/O's, planning & budgeting, etc) • HR	Needs to be updated.	H
7	CodeWrite	PC based editor	Allows connect to source	H
	Development quotes	Planning, approval and management	The routing process is very long. Does not work properly. It should be only a formality. If the projects are defined and approved in advance, before starting a new release, the approval should be done either by the VP Development or the R&D Manager	H
12	***EliteSeries***	To research.		H
11	***EliteClient***	Run Elite / Telnet Access	Allows anything from the screen to be copied "with attributes" to WORD. Extremely helpful in user interface design. Makes it easy to dial in to customer sites.	H
2	Email	Written communication of short points requiring clarification.	Effective – Reply may be added as an appendix or supporting document to the design. However can be used too frequently rather than actually talking to	H

			someone.	
	Excel	Can be used for a multitude of purposes. eg: Planning, statistics, reports, budgeting, test plans etc.		M
2	Flowcharter	Produce Diagrams, flow charts, etc	Good for process flows, entity relationships, and general business diagrams. (A bit buggy at times).	M
3	Framemaker	To generate reference guides & books	Good for reference material development. (A bit buggy)	M
4	Internet Explorer or Netscape Navigator	Research and information gathering	Hit and miss source of information. Many web sites are sales and marketing oriented.	M
	Informix Tools	Allows for the development of programs and the maintenance of the database.	No GUI Interface.	H
	MS Project	Planning	Has limitations. We should invest time to find out if better tools exist on the market.	H
1	Phone	Verbal communication	Works well. All decisions should be followed-up with written confirmation.	H
13	Printed Data Model	Data Model	• Use it many times per day • "Big Picture" is always on hand. • Printed copy is grossly out of date. • No access to on-line version.	H
14	PVCS	Version Control	• Not all functionality have been implemented with our development environment at 100% compatibility. • Not tested on smaller projects before being used on larger ones.	M
5	Scripts	Numerous scripts from setting up the environment to scripts used for releasing etc…	Not properly documented. Many problems (They need to be updated in many cases).	H
3	S-Designer	ER Diagrams and database definitions	Currently only available to DBA - could save redundant work. Not used to its full	H

			extent.	
	Time job	To verify if R&D people have entered their time against the proper job number. Used for billing & month-end reports etc… Record all S/W and documentation problems found - all depts except for support	Not very user friendly. SCM & TJ → two different systems in the same company having the same purpose to track problem	H
10	vi	Edit UNIX files.	Very technical. Not user friendly. To be phased out by CodeWrite.	M
4	Word	Create Windows based documents such as anal[illegible]design, quotes, as a means of communication (letters, reports, etc).	• Allows for standard documents to be created that include the basic screen, validation, help and processing to be communicated to the programmer. • Used for but Not effective for data modeling.	H

5. CONSTRAINTS

The constraint table is used to identify the constraints that have been imposed on the process.

Constraints ...

#	Constraint	Impact
1	Time – pressure to deliver product and knowledge. Release deadline or resource availability.	H
2	Limited to tools that document the thought process rather than helping to define the business process. (i.e.: CASE Tools).	H
3	Must adhere to the standard approval cycle. (Quotes, analysis, design, etc.)	M
4	Limited exposure to EliteView, EliteQ and EliteWMS – must await resource	M
5	Concurrent Projects. Makes the development cycle longer. Design is obsolete before it is ever approved.	M
6	Not having the knowledge to answer the question. Training issue.	M
7	Product roll-out not effective. (Knowledge transfer, transitions period).	H
8	Head count for the department. Has an impact on the work that could be delivered by the group.	H
9	Not having a list of features before starting a new version with a fixed delivery date.	H
10	No clear direction/vision as it relates to corporate mandate. (Short/long term vision, corporate philosophy, technology)	H
11	Departmental Budgets	M
12	Resource availability. (Both time as well as skill sets).	H
13	Translating Spanish last → lack of continuity, difficult to plan other releases that are moving forward	H
14	Effort permitted is often limited due to predefined time/cost constraints - jeopardizing best possible solution	M
15	Data Modeling tool is not available to Analysts to capture database design concepts and definitions	L
16	Lack of material to facilitate knowledge transfer.	H
17	High Ramp-up costs Learning Curve.	H
18	Involvement in custom development projects.	H
19	Responsiveness to hardware / network related issues.	H
20	Source conflicts.	H

6. POSITIVE ASPECTS

The following table is used to capture all the positive aspects of the current process, in terms of anything that is working well and should potentially remain unchanged.

What works well ...

#	Positive Aspect	Comments	Value
1.	Quality	Multiple levels of approval helps ensure quality.	H
2.	Quality	Test plan helps pass knowledge. QC forms, Development Check List, etc.	M
3.	Standardization	Ensures that all documents/programs have a similar 'look and feel' for presentation to internal and external sources.	H
4.	System Documentation	The level of system documentation is superior to what it has been in the past.	H
5.	Central repository for tracking of bugs/enhancements.	Provides a common point of reference for future evaluation of functionality, enhancements and bug fixes (ensures that nothing is missed).	H
6.	Development Documentation	Important to document the functionality that is being developed. Used as a means of documenting the options, decisions, restrictions, etc.	H
7.	Support	We provide good support to other groups in terms of responsiveness, attitude and quality.	H
8.	Planning	A master project plan is maintained.	H
9.	R&D forms (quotes & PO)	Ensure better planning.	H
10.	Tracking origin of bug	5 / 5Q / 5C - Is important to track origin - as a method of evaluating performance	M
11.	QC/CR by other resources	Very important and works well; nothing goes out without formal QC → better product	H
12.	Translation maintains both maintenance release in parallel with current release	Reduce number of inconsistencies between releases.	M
13.	In house scripts	Help to improve productivity, reduce redundancy, etc. Eg.: The transrel utility creates a back-up of current option and releases. This allows the user to run the compfh utility to get a report of what has been altered since last release.	H

14.	Translation procedures well documented	All procedures are very well documented and are an easy reference for anyone that does not know how things work	H
15.	Time tracking	We do a good job of tracking time spent on each phase	H
16.	Estimation review process	Having all estimates reviewed by a single resource promotes consistency and accuracy	M
17.	Kick-off Sessions	The concept is great - it allows for a cleaner hand-off.	H

7. NEGATIVE ASPECTS

The following table is used to capture all the negative aspects of the current process, in terms of anything that is not working well and should reviewed for potential changes. This table is not used to capture ideas to improve the problem, rather, it is used to simply identify the issue.

What doesn't work well ...

#	Negative Aspect	Comments	Impact
1	Time Job	Difficult to extract information that is truly valuable to improving design/quotes	H
2	PVCS	Not fully integrated!	M
3	Review/Approval Time	Approval Hand-off's are slow unless the project has "top priority".	H
4	ABC-Flowcharter	"Buggy" interface to word is a bit of an annoyance.	L
5	Test Plans	Long to prepare – questionable value gain for some projects. QA does not have responsibility for review/creation/input until after project completion	H
6	Design Document	Time-consuming for easier modifications. Evaluate level of detail.	M
7	Product Standardization	Happens too late in the process.	H
8	Product Translation	Happens too late in the process.	H
9	QA Timing	QA's involvement is too late in the process.	M
10	DB Model	Printed model out of date for over 1 year – lack of access to a modeling tool	M
11	Tracking tools	Support and other departments not using the same tool nor is it being used consistently across departments.	L
12	Definition of the future version	Not having a defined plan for the next versions.	H
13	Training	At the end of the version, training always suffers. We do not having enough time to prepare for and do the roll-out to other departments.	H
14	Job approval	The approval process is too long.	L
15	Customer contracts	We don't follow-up effectively on our customer commitments.	H
16	Communication	We need to make other teams aware of what we do and be aware of what others are also doing.	H
17	Code review	Needs to be done for bug fixes.	M
18	Time Job Entry (Bugs)	If entry not correctly done - may never be seen by R&D. Too complex. Too many codes, undocumented rules, lack of validations, etc.	H
19	Time Sheet	User interface very user un-friendly.	M

20	Allocations of resources	Based on availability and time constraints as opposed to best suited resource.	H
21	Translation Procedures	Too many procedures in the translation department. Very difficult to follow everything every step of the way - does not let the translator feel very creative	H
22	R&D Administrative Procedures	Simply and document the procedures to be followed.	L
23	Timing of Translation	Need more time to complete a release after release management and R&D have finished development; Pressure too high and too little time; have to compromise certain rules to get the job done; leaves room for error.	H
24	Integration Testing	Often compromised as a result of delivery deadlines.	H
25	Performance Testing	We do not do adequate performance / load testing.	H
26	Time delay / message creation	Has impact on development due to delay in creating, standardizing and translating/QC'ing the said messages; programmer must re-test after going back to his messages and incorporating them in the code.	H
27	Collaborative Work Groups	Too many independent hand-offs without enough collaborative work groups/teams. Lack of knowledge related to continuity, assumptions made during previous phases of development, etc.	H
28	Business Requirements insufficient or missing	Occasionally we will perform analysis without a formal Business Requirements specification prepared.	H
29	Future project resources are not involved early	1. We do not (or cannot) define project teams far enough in advance, to ensure that they are included as a part of the brainstorming phases. 2. We do not always involve subject matter experts (whether or not they are to be on the project) in brainstorming phases.	M
30	Partial data dictionary		H
31	Insufficient system documentation	System hierarchy, process flows, entity relationships, data flows, script documentation are missing	H
32	Tools	Analysts are using Flowcharter to design ER diagrams	M
33	Frustration level due to incomplete solutions	Occasionally Analyst will be frustrated because we are not permitted to flush out the best possible solution - we are mandated to "keep things simple"	M
34	Analysis and Design documents	Documents do not get updated consistently when design changes are made causing problems in the hand-off to QA, as they cannot rely on these documents to be current.	M
35	Lack of adequate test database, hardware and tools.	Need more engineering tools rather than documentation tools.	H
36	Prioritization	Bug fixing on closed projects are done at the end of the release.	L

37	Roles	Resources are not used to their fullest potential	M

8. CRITICAL SUCCESS FACTORS

The following table represents those factors which are deemed to be absolutely critical to the success of the process and deliverables alike. In other words, if these factors are not applied, we are sure to fail.

Critical Success Factors ...

#	Critical Success Factor	Impact of NOT Implementing
1	Resource Availability • Expertise (WMS, EliteView, EliteQ, EDI, etc.) • Focus on R&D (Generic product). • Sufficient head count.	Efficiency & Quality: Longer effort and potentially less quality solution
2	Reduce the duration of the review / approval process. (Not the effort involved).	Delays.
3	Involve the right people at the right time.	Delays, problems with quality.
4	Tools • Advanced analysis, design and development and debugging tools.	Quality and time to market is not improved.
5	Ensure continuous learning. • New business concepts • New tools / software / technology	Lack of efficiency and expertise.
6	Knowledge Transfer (training) • Timeliness • Audience (R&D, C/E, Support, Customers)	Lack of focus. i.e.: Interruption level will be much higher. Quality of future modifications will not be as high.
7	Effective Communication • People must be made aware of what we are doing. • We must be aware of their requirements (product related). • We must be aware of customer commitments. • Changes to planned features.	Unrealized expectations.
8	R&D (Product Management) must be involved in all product related commitments. • R&D must be privy to customer commitments. • Must have clear objectives for product development.	Unrealized expectations.
9	Planning • Establish a more concrete plan	Without having a plan, it becomes difficult to predict when and how the

	• Definition of feature release • Proper time allocation	group could deliver.

9. FUTURE STATE ISSUES

1.	Standardizing Design docs	Would be excellent practice as the help would be formatted and set up in word prior to starting the development → the help is then already standardized and could go directly to translation. Less bottlenecks and more efficient! (Not yet implemented)	H

ANNEXE 4

Rapport - Équipe d'assurance qualité

TECSYS

RESEARCH & DEVELOPMENT

Business Process Design

Current State Analysis
Quality Assurance

August 28, 1999

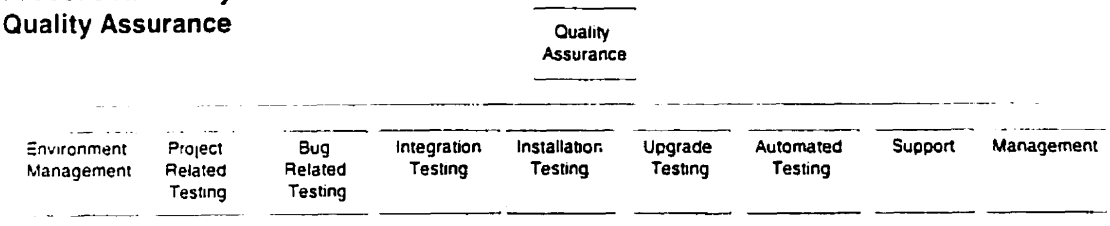
TECSYS INC.
1840 Trans Canada
Dorval, Quebec
H9P 1H7
Tel.: (514) 333-0000
Fax: (514) 333-0109

Table of Contents

1. PROCESS HIERARCHY

1.1 HIGH LEVEL PROCESS HIERARCHY

Process Hierarchy
Quality Assurance



QUALITY ASSURANCE

1.2. PROCESS HIERARCHY - TEXTUAL

1.2.1. Thread 1 - Environment Management

EBP 1. Trigger → **Daily Release Log**
- Created & printed automatically overnight.
- The following tasks are performed by the "QA Coordinator".

EBP 2. Verify report to ensure that the programs promoted to QA were compiled without errors.

EBP 3. Contact the SE department to investigate, if there are errors. *
- SE investigates the error and corrects the problem manually.
- SE notifies QA coordinator when correction is complete.

EBP 4. Copy the release log information to the appropriate Excel log (6.0, 6.1, 6.2).

EBP 5. Verify if any of the bug fixes are customer calls (TAR) and highlight these on the log.
- Print the SCM summary report for calls transferred to R&D
- Verify if any bug fixes on the release log are in the SCM report and flag these bug fixes as top priority.

EBP 6. Verify that the priority of all calls on the release log is indicated on the Excel log.
- if the priority is not indicated, assign from Time Job

EBP 7. Assign bugs to the QA analyst(s), if necessary.
- If a rejected call has been re-released, the QA analyst that originally tested the call will re-test.
- A WMS related call is assigned to specific analyst.
- A specific call that Support or R&D is awaiting completion.

Note: Generally QA analysts self-assign calls based on priority in the Excel log.

EBP 8. Log time spent in Time Job.

EBP 9. Trigger ← Excel log priorities and exceptions have been assigned

* Conditional items that are not always performed.

1.2.2. Thread 2 - Project Related Testing

EBP 1. Trigger → **Project is promoted to QA**

EBP 2. QA Manager assigns QA analyst(s) to the project.

EBP 3. The assigned QA analyst(s) reads the documents.
- Business Requirements document (if it exists).
- Analysis document.
- Design document.

EBP 4. Attend kick-off meeting (between the R&D analyst/designer and the QA analyst(s)).
- Review documents (identified above)
- The job code is provided to the QA analyst
- A copy of the test plan, if available is given to the QA analyst *
- Explanation of functionality/limitations
- Discuss the process flow
- Review the Data Model
- Hands-on introduction to the new functionality

EBP 5. The test plan prepared by the R&D is reviewed and enhanced by the QA analyst. *
- If a test plan is not available, the QA analyst prepares a test plan.
- The test plan is then reviewed with the R&D analyst/designer/programmer for completeness.

EBP 6. Unit Testing of the project is performed.
- Familiarization with the new functionality
- Prepare and enter test data
- Verify that programs/functionality conform to the design specifications
- Data integrity testing
- Functionality testing
- Check usability
- Verify input validations
- Verify standard function keys
- Check if it is standard with the rest of Elite
- Check help file modifications (usually prior to EMD standardization and translation)

EBP 7. Test plan is revised/enhanced. Test points are added or altered. Results are recorded.

EBP 8. The bugs and/or findings are discussed with the QA coordinator, if necessary.
- If the information is insufficient, the R&D Analyst/Designer/Programmer is consulted.
- If there are issues that affect the functionality and were not covered in the design, Product Direction is consulted.

EBP 9. The bugs are recorded in Time Job.
- Entered as type "QP" and the project name is entered in the P/O Number field.

EBP 10. The QA analyst notifies the R&D project leader/analyst so that they are aware of the calls that were entered.

EBP 11. Testing continues, if possible.

EBP 12. R&D corrects the bug or justifies the functionality.
- The project leader notifies QA once the call is set to 'QA' status.
- R&D promotes (tool: PVCS).

EBP 13. The Excel log is updated (see Environment Management)

EPB 14. The QA analyst tests the bugs/corrections (re-evaluating EBP 6 above).

EPB 15. Accept: Close call in Time Job, recording notes.

EPB 16. Reject:
- The QA analyst reviews the rejected bug with QA coordinator, if necessary.
- Then it is reviewed with R&D, if necessary.
- The bug is put to "RJ" status in Time Job, and the reasons for the rejection are recorded.

EPB 17. The QA analyst notifies the R&D project leader/analyst so that they are aware of the call that was rejected.

EPB 18. Integration Testing
- A complete run-through is made on the project – all programs, all options.
- Functionality testing
- Data integrity testing
- Check if it is standard with the rest of Elite
- Carry out transactions from beginning to end for the process flow, verifying G/L entries, audit trails, costs, etc.

EPB 19. Revise test plan, if necessary. Test points are added or altered. Results are recorded.

EPB 20. Log bugs in Time Job.
- Repeat steps EBP 10 through EBP 19 until bug correction is accepted by QA analyst.

EPB 21. Log time (in Time Job) against the job number created by R&D.

EPB 22. Trigger ← Notify the R&D manager and QA Manager stating that the project can be closed, once all the bugs have been closed and testing is complete.

1.2.3. Thread 3 - Bug Related Testing

EBP 1. Trigger → **Bug Fix is promoted to QA**

EBP 2. The Excel log is used to determine the testing priorities.
- customer calls (marked TAR)
- calls in order of priority (1-critical,2-high,3-medium,4-low)
- QA analyst self-assigns work from this log, if not specifically assigned tasks by QA Manager or QA coordinator.

EBP 3. The QA analyst marks their initial in the "Tester's Name" column on the Excel log.

EBP 4. The QA analyst marks their initial in the "Assigned to" field of the call in Time Job and prints out the call.

EBP 5. If the call refers to a customer call, the QA Coordinator prints the TAR from SCM (Service Call Management).

EBP 6. The call is read and reviewed by the QA Analyst.
- If further information is required as to scenarios to test, release problems, questionable results, the QA Coordinator is consulted.
- If insufficient, then the questions are directed to the individual that entered the call, the programmer or the RM manager.

EBP 7. An "informal" test plan is used depending on the type and scope of the bug call.

EBP 8. Unit testing of the bug is performed.
- Test in previous version (if applicable) to see the impact of the call.
- Prepare and enter test data.
- Test that the fix corrects the stated problem
- Test that the fix is complete by performing other test scenarios
- Data integrity testing

EPB 9. If the bug cannot be duplicated by RM, but can be duplicated by QA:
- QA Analyst updates Time Job, indicating the steps required to duplicate the bug.
- The call status is put back to new (NE)
- The initials of the original programmer are entered in the Assigned to field
- QA Analyst emails the Programmer, QA coordinator and RM Manager

EBP10. Integration testing of the bug is performed.
- Carry out transactions from beginning to end for the process flow
- Data integrity testing
- Functionality testing
- Check if it is standard with the rest of Elite

EPB 11. If there are issues that affect the functionality and were not addressed in the bug fix, Product Direction is consulted.

EPB 12. Accept:
- The QA analyst closes the call in Time Job and records their notes.
- The Excel log is updated to indicate that the call has been tested.

EPB 13. Reject:
- The QA analyst reviews the rejected bug with QA Coordinator, if necessary.
- It is reviewed with the programmer, if necessary.
- The bug is put to "RJ" status in Time Job, and the reasons for the rejection are recorded.
- The Excel log is updated to indicate that the call has been rejected.
- An email is sent to the Programmer , RM Manager, and QA Coordinator.

EPB 14. Other bugs encountered in the testing which are not related to the call being tested or which do not justify rejecting that call, are entered in Time Job.

EPB 15. Log time spent testing the call in Time Job.

EPB 16. A hard copy of the call (accepted or rejected) is filed by the QA Analyst, to be reviewed by the QA coordinator.

EPB 17. The calls are reviewed by the QA Coordinator, to ensure that testing is complete.

EPB 18. If the call is a TAR, Service Call Management System is accessed and the R&D status for the call is put to "Closed"

EPB 19. Trigger ← The call is closed in Time Job by the QA analyst, for the release *version* that was tested.
- Once all versions have been tested the entire call is closed.

1.2.4. Thread 4 - Integration Testing

EBP 1. Trigger → **All projects and related bugs (priority 1 and 2) have passed QA.**

EBP 2. Usually a Project Leader is assigned, who
- Is closely involved in the testing.
- Has the business background to assess testing results in terms of acceptability.

EBP 3. The QA team prepares test plans for the integration test.

EBP 4. A kick-off meeting is held with the individuals involved in the test.
- The objectives of the test are reviewed.
- The testing guidelines are reviewed.
- The work teams are assigned.
- The test plans to be used in the process are distributed.
- The procedure for using the test plans is reviewed.
- The procedure of how to log time and enter bugs is reviewed.

EBP 5. Integration Testing
- Perform a run-through test of the projects and the interdependencies between them.
- Functionality testing.
- Data integrity testing.
- Carry out transactions from beginning to end for the process flow verifying G/L entries, audit trails, costs, etc.
- Example: Generate an order in Elite, ensure that it is transferred to WMS and processed correctly within WMS (picked, packed, shipped, inventory adjusted). Ensure that it is transferred back to Elite and generates correct financials.

EBP 6. Log bugs in Time Job.
- Entered as type "QP" with a specific P/O Number (INTEGRATION).

EBP 7. The Project Leader assesses the impact of the bugs found.
- Materiality of the bug.
- Acceptance risk.
- Ability of still meeting delivery date.

EBP 8. Notify R&D Project Leader when priority 1 and 2 calls have been entered.

EBP 9. Test the bug corrections.

EBP 10. Accept the call and close the call in Time Job. Or reject the call and put into "RJ" status in Time Job, recording the reasons for the rejection.

EBP 11. The Project Leader provides the QA Manager with an ongoing status of the results of integration test.
- Functionality compromises are considered and their impact on acceptance risk are evaluated.

EBP 12. Log time in Time Job.

EBP 13. Trigger ← Notify QA Manager stating that the integration test is complete.

1.2.5. Thread 5 - Installation Testing

EBP 1. Trigger → **All programs being part of a feature or maintenance release, have passed QA.**

EBP 2. QA Coordinator notifies SE to release all the programs, libraries, and scripts from QA's testing environment, to production.

EBP 3. Kick-off with SE to determine which combinations of installation tests to run, and how many. This is also reviewed with the Support Manager.
- New installations versus upgrades
- On-line engine versus standard engine
- With source and without source
- In RDS or in C-compiled

EBP 4. After the specified installation tests are defined, SE begins their preparation.
- SE cuts master, etc.

EBP 5. QA Coordinator is notified of the installation directories that have been created for testing.

EBP 6. The QA Analyst accesses the installation directory defined by SE and begins the installation test by following the installation procedure.
Example:
- Would you like to review the installation help?
- Are you using Informix SE?
- Install English only?
- Install live database?
- Install training database?

EBP 7. When the installation is complete, the installation log is verified for any errors.

EBP 8. Errors are reported to SE.
- Depending on the error(s) found, testing may continue.

EBP 9. SE creates an environment accessing the training database that was loaded (if necessary).

EBP 10 QA runs tests in the environment on major programs.
- Orders are entered and followed through to invoice and cash receipt.
- Purchase orders are entered and followed to receipt and cash disbursement
- Inventory adjustments are entered
- The on-hand, costs, movement, and journal entries are verified

EBP 11. The installation test is run through the automated 'Winrunner' installation test, (if available).
- The Winrunner test ensures that all 'configured' programs exist and they simply 'load' successfully.
- The results are verified by QA.

Note that Winrunner has not been updated since 6.0.3.

EBP 12. Trigger ← When all installation tests are complete (and bugs corrected), Support, PS, RM, and SE are notified that QA is complete.

1.2.6. Thread 6 - Upgrade Testing

EBP 1. Trigger → **All programs and projects part of a feature release have passed QA.**

EBP 2. A database is required .
- A customer database would be preferable; a customer that does not have modifications.
- Or a demo/training database that replicates a target customer.

EBP 3. The SE department loads the customer database with the corresponding programs.

EBP 4. The database comparison utility is used to run a comparison between the customer's database and the generic standard version.

EBP 5. The SE department prepares an environment for QA for testing purposes accessing the current database and current programs.

EBP 6. QA prints key reports from the various Elite modules: trial balance, aging, order register, shipment register, etc.

EBP 7. The upgrade program is run to upgrade the database from the existing version to the next version.

EBP 8. The database comparison utility is used to run a comparison between the upgraded database and the generic standard version.

EPB 9. When applicable, (ie when source is included in the upgrade), the upgraded source files are compared against the master to ensure that all sources were upgraded properly.

EPB 10. The SE department prepares an environment for QA accessing the upgraded database and the new programs.

EBP 11. QA verifies that the data was correctly transferred to the upgraded database.
- Major tables are checked.
- sqls are run in both environments to ensure that the data was correctly transferred.
- The upgrade program determines how to populate the new tables and/or fields. A design document exists. (This is done prior to the upgrade).

EBP 12. The reports that were run in the old version should be re-run in the upgraded version for comparison purposes.

EBP 13. New transactions are entered in the upgraded version to ensure that the tables are correctly populated.
- Information that existed in the old version should also be used. Example: Process a return and an invoice adjustment that refers to an invoice that existed in the old version.

EBP 14. Trigger ← When the upgrade tests are complete, Support, PS, RM, and SE are notified that QA is complete.

1.2.7. Thread 7 - Automated Testing

EBP 1. Trigger → **Maintenance Release**

EBP 2. There are 3 kinds of automated tests:
- Installation Tests
- Exhaustive Unit Tests (EUT)
- Data Integrity Tests (DIT)

EBP 3. Running the tests involves a controlled database which is called the Initial Set of Conditions (ISOC).
- Installation tests can be run on a default database.
- EUTs and DITs must be run on the ISOC database.

EBP 4. The installation test is the first pass to make sure the basic functionality exists.
- Ensures that all the functions are there and they all load successfully.
- The results are output to a report.

EBP 5. The exhaustive unit test ensures that certain functions of a program produces the expected results.
- There is 1 EUT script per option (screen, report, batch) within Elite.
- The EUT script can be run individually to test a specific program or in batch mode to test all the programs of a module.
- The EUT scripts test every function of the ring menu, every function key and every field.
- The EUT scripts load the program, create, update and delete records.

EBP 6. The data integrity test focuses on accounting integrity and statistical integrity.
- The DITs is a 3 phase process. The first phase is to set up the static tables and conditions according to a business scenario.
- Second phase is entering the business transactions per the scenario.
- Third phase is validating the results against the expected results.

1.2.8. Thread 8 - Support

EBP 1. Trigger → **Help!**

EBP 2. Support to Documentation dept.

- The file containing the copies of the corrected calls is provided to EMD, on an on-going basis for the preparation of Release Notes.
- Once the Release Notes are complete, they are reviewed by the QA Coordinator.
- Review of maintenance release notes
- Review of Elite documentation (user guides, reference guides)
- Review of WMS documentation (WMS user guides, interface documentation)

EBP 3. Support to Professional Services

- Questions, clarification, or explanations about Elite or WMS functionality

EBP 4. Support to Sales/Marketing (Marc Wulfraat)

- Questions, clarification, or explanations about Elite or WMS functionality

EBP 5. Support to R&D

- Functionality in relation to WMS

EPB 6. Support to Support dept.

- Functionality and documentation in relation to WMS

EPB 7. Internal Support

- Providing support within the QA dept.

EBP 8. Trigger ← Solicitor is informed about current processes/functionality/limitations.

1.2.9. Thread 9 - Management

- Feature Release Related
  - Participate in Product Planning sessions
    - identifies scope of feature release
    - high level definition of feature release projects
  - Status meetings with development groups (R&D, SE, NTR, RM)
    - status on planned projects
    - review R&D plan for changed project deliveries to QA
    - confirm delivery dates to QA
    - identify training required by QA
    - resolve planning conflicts
  - QA Planning
    - based on projects to be included in Feature Release
    - generate QA plan for Feature Release and its projects
    - ensure QA resource vacations and time off included in plan
    - assign resources, dependent on skill and schedule time
    - communicate task assignments to QA resources
    - determine Time Job codes to use for recording bugs found in a project
    - ongoing plan updates based on day to day issues, new tasks, delays, etc.
    - review of Open Summary Task Report
      - ensure QA accepted projects closed
      - confirm all projects planned by development groups are included in QA plan
    - ongoing communication of status to QA team
    - ongoing communication of QA deliveries to Tecsys management
  - Individual Projects
    - communicate to QA resources delivery of project out of R&D to QA
    - communicate to QA resources delivery out of QA
    - assign projects to QA resources
    - weekly meeting for project updates from QA resources
    - bi-weekly QA team meeting to review feature release plan and to share project updates amongst team
    - project status from QA resources
      - current status?
      - on track to meet deadline?
      - what are the issues
      - can more resources assist in meeting deadline?
      - what resources are available to assist?
      - impacts on other planned or in process QA projects
    - resolve or escalate problem(s)
    - communicate delays and issues to Tecsys management
    - performance monitoring
      - are tasks taking longer than expected? why?

- Management of Testing Environments
    - Responsible for maintaining master QA database.
    - Responsible for coordinating effort when specific testing environments are required.
        - Testing EliteQ and EliteView.
        - Testing Elite with the WMS Interface.
        - Testing Elite without the WMS Interface.

- Training
    - identify training requirements based on projects hitting QA
    - review QA resources current skills inventory
    - determine training candidates
        - share new skills opportunity across QA team
        - based on project schedule who are best suited
        - confirm QA resource interested in training opportunity
    - determine if courses are available
    - if not, raise training need with development team responsible for new feature/technology
    - coordinate training schedule with Tecsys Training Coordinator and QA resources attending
- Administrative tasks
    - Billing
        - Approval of inter group billings
        - Approval of QA overtime for payment
        - Verification of the month end reports

2. Business Rules

Business Rules define the rules that govern the process.

Business Rules ...

#	Business Rule	Enforced	Effective
1	Programs promoted to QA must be compiled in our environment overnight.	Y	H
2	Daily release log must be printed automatically indicating if compile was successful.	Y	H
3	Release log information must be copied into Excel log.	Y	M
4	Must specify on the Excel log if the bug fix is a customer call (TAR) and the call's priority.	Y	M
5	All bugs recorded in Time Job must be prioritized.	Y	H
6	Bugs mus. be QA'ed according to TARs and then by priority.	Y	H
7	Excel log must be updated throughout life cycle of testing a bug.	Y	H
8	Time Job must be updated throughout life cycle of testing a bug.	Y	H
9	Bugs found are recorded in time job system.	Y	H
10	All bugs put to "QA" status must be tested.	Y	H
11	Bugs must be closed by QA only.	Y	H
12	All calls that are TARs must be closed in SCM.	Y	M
13	Bugs found in a project for a feature release must be recorded as type "QP".	Y	H
14	Formal Business Requirements doc , Analysis doc & Design doc must exist	N	H
15	A Job Code must exist to record time against .	Y	H
16	QA Kick-off session must be conducted.	Y	H
17	Test plans must exist or else be created for a project.	Y	H
18	Test results must be documented for projects and bugs.	Y	H
19	High risk areas are tested first and then lower risk areas.	Y	H
20	Time spent on project and re-testing of project bugs are recorded against the job code.	Y	H
21	R&D and QA Manager must be informed when a project can be closed.	Y	M
22	RM Manager, programmer, and QA Coordinator must be informed when bug fix is rejected.	Y	M
23	Status updates must be provided on a timely basis to QA Manager.	Y	M
24	Quality is the driving force to ensure optimal business functionality and completeness of product.	Y	H

3. Performance Factors

Performance Factors define how the process is to be measured in specific terms, and captures whether the measurement is actually tracked in today's operations, along with it's value.

Performance Measures ...

#	Category	Performance Description	Measurement	Track	Value
1	Quality	Assessing quality of projects, quality of bug fixes, and overall product quality.	Tracking of bugs found by project, by priority for feature releases. Tracking of rejected calls within a maintenance release. Tracking of bugs, by priority, during integration testing.	Y	H
2	Efficiency/Timeliness	Planning/Scheduling accuracy	Tracking of actual delivery to QA by R&D versus planned delivery. Tracking of actual delivery out of QA versus planned delivery.	N	H

4. Tools

The purpose of the Tool table is to define the various tools and or systems that are required to facilitate the process.

Tools ...

#	Tool	Purpose / Use	Comments	Value
1	Ms Project	QA Planning	Unable to split tasks when interruptions in work occur. Effective for QA planning.	H
2	Word	QA Communications/Guidelines To document test plans	Effective for informal processes. Effective depending on the type of testing.	H
3	Excel	To document test plans To track the calls in QA	Effective depending on the type of testing.	M
4	TimeJob	Recording time. Recording bugs.	Shortcoming exist and product could be significantly enhanced.	H
5	SCM	List of customer calls in R&D Obtain further information on calls Close R&D calls once QA'ed	Having to go to 2 systems for bug information.	M
6	Winrunner	Automated testing tool	Not kept up-to-date	H
7	MS - Exchange	Communication between departments and within QA.	Could be used more effectively.	M
8	Regular QA Status Meetings	To keep the QA team up-to-date.	Keeps everyone informed.	M

5. Constraints

The constraint table is used to identify the constraints that have been imposed on the process.

Constraints ...

#	Constraint	Impact
1	Restricted mandate of QA Manager to effectively manage team due to interim position.	H
2	Ineffective communication between development teams, QA occasionally surprised by new projects or delays.	M
3	Lack of QA tools (automated and other)	H
4	Tight scheduling timelines - product quality versus delivery on time	H
5	Lack of technical expertise	M
6	Actual QA resources versus the workload.	H
7	For knowledge transfer and input, QA is not involved early enough in the process.	H

6. Positive Aspects

The following table is used to capture all the positive aspects of the current process, in terms of anything that is working well and should potentially remain unchanged.

What works well ...

#	Positive Aspect	Comments	Value
1	Support Quality	QA team lives, breathes and believes in Quality.	H
2	Communication	Internal group communication is strong.	H
3	Knowledge	Knowledge is shared among the group members.	H
4	Kick-off Sessions	Kick-off sessions with R&D work well.	H
5	Quality Tracking	Tracking of bugs found, by priority while testing a project.	H
6	Test Plans	Use of test plans for projects, minor enhancements, and major bugs.	H
7	Testing Skills	Ability to focus testing the higher risk areas.	H
8	Call Assessment	Ability of evaluating the materiality and severity of bugs .	H
9	Call Review	All calls are reviewed by the QA Coordinator for completeness.	M
10	Integration Testing	Ensures the overall quality and completeness of the product.	H
11	Team Work	There is good rapport within the QA team and with R&D, and RM	H
12	Sharing Resources	Resources provided to QA from R&D in order to meet specific deadlines.	H
13	Planning and Scheduling	Weekly status meetings with all groups to plan for projects.	H
14	Mentorship	Working as a team within QA, helping each other.	H

7. Negative Aspects

The following table is used to capture all the negative aspects of the current process, in terms of anything that is not working well and should reviewed for potential changes. This table is not used to capture ideas to improve the problem, rather, it is used to simply identify the issue.

What doesn't work well ...

#	Negative Aspect	Comments	Impact
1	QA Development Review Sessions	We do not have enough QA review sessions/involvement throughout the development process to brainstorm ideas and review progress – we tend to wait until the end. QA resources are not involved early enough.	H
2	Frustration level due to past history	QA frustrated due to lack of mandate and direction from Teesys.	H
3	Time Job	No ownership, no security, no validations , no standards , not effective as a bug tracking tool.	H
4	Redundant Bug Tracking	Lack of development ownership results in QA creating redundant logs and reports to ensure bug fixes are delivered to QA.	M
5	QA Testing Tools	Only 1 out dated automated testing tools in place, no other QA automated testing tools available.	H
6	Documented Processes	Need to document QA procedures, standards, testing checklists	H
7	Resources	Allocation based on availability and not "right" individual. Resources not being used to their fullest potential.	H
8	Planning versus Scheduling	Too much time spent on Scheduling which constantly changes. More time should be spent on planning .	H
9	WMS and VCS	WMS is not on PVCS, everything is done manually.	H
10	QA Testing Scope	QA identifying bugs outside of the scope, frustrated when challenged to open new call and not reject an existing call. QA testing in previous release to see if bug already existed. No guideline exists for determining rejection of calls.	M
11	Problem description on bugs not clearly stated.	The problem description of the bugs is not always descriptive in terms of explaining what the problem is and how to duplicate it. QA spends a lot of time trying to determine the problem what the problem is in order to test the fix.	H
12	Responsibilty of compilation report	QA is responsible for validating the nightly compilation report, to ensure that all programs that were promoted to QA compiled correctly.	H
13	Lack of standard test plans	Test plans recreated for each project, no test plan templates available to expedite	H

		the process.	
14	Integration Test Plans	No standard test plans exist, create for 6.1 but not saved as standard templates.	H
15	Lack of Test Data	No tools exist for creation of test data, created manually by project.	H
16	Lack of RM Test Plans	No test plans [illegible] QA from RM on major fixes and enhancement calls.	M
17	Lack of training	No outside training on testing approaches to improve testing methodologies.	M
18	Lack of communication and technical support.	Technical changes occasionally executed without taking QA issues into consideration.	M
19	Lack of effective master testing database.	The master testing database has not been maintained and is not up-to-date.	M
20	Performance Testing	QA does not conduct volume/performance testing.	H
21	Documentation	Business Requirement and Analysis documents don't always exist.	H
22	Documentation	Analysis and Design documents are not kept up-to-date with the current changes.	H

8. Critical Success Factors

The following table represents those factors, which are deemed to be absolutely critical to the success of the process and deliverables alike. In other words, if these factors are not applied, we are sure to fail.

Critical Success Factors ...

#	Critical Success Factor	Impact of NOT Implementing
1	Resources • sufficient resources • knowledgeable of different Elite products • assign the "right" resource for each project	Quality suffers
2	Tools • bug tracking system • testing tools	Lack of efficiency
3	Planning and Scheduling • schedule always changing • should be more concrete • should be more planning and less scheduling	Lack of efficiency

ANNEXE 5

Rapport - Équipe de déploiement des versions

TECSYS

RESEARCH & DEVELOPMENT

Business Process Design

Current State Analysis
Product Release

Final Copy

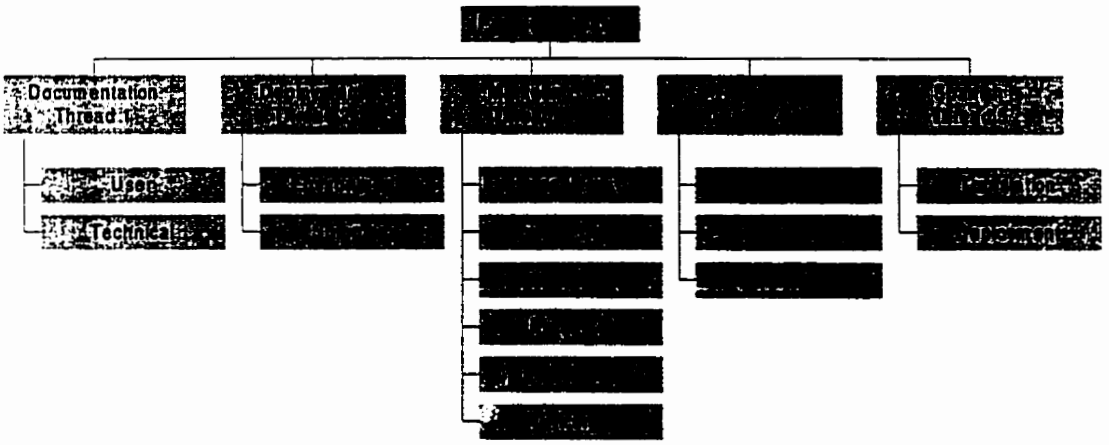
August 28th 1999

TECSYS INC.
1840 Trans Canada
Dorval, Quebec
H9P 1H7
Tel.: (514) 333-0000
Fax: (514) 333-0109

Table of Contents

1. PROCESS HIERARCHY

1.1. HIGH LEVEL PROCESS HIERARCHY



1.2.

PROCESS HIERARCHY - TEXTUAL

1.2.1. Thread 1 - Documentation - User Doc. - English User Guides

Note: Process used to be scheduled and was working well. A lack of resources means that this project has fallen behind. Currently, the guides are not being maintained and new work is not being scheduled.

1.2.2. Thread 1 - Documentation - User Doc. - English Start-up Guides

Note: Process used to be scheduled. A lack of resources means that this project has fallen behind. Currently, the guides are not being maintained and new work is not being scheduled.

1.2.3. Thread 1 - Documentation - User Doc. - English Financial Control Guides

Note: Process used to be scheduled. A lack of resources means that this project has fallen behind. Currently, the guides are not being maintained and new work is not being scheduled.

1.2.4. Thread 1 - Documentation - User Doc. - English Misc. Guides

Trigger → A request is made to create/update and publish a new guide (ex. ***EliteSeries*** WMS procedural work around documents).

EBP 1. Open a job code

EBP 2. Allocate resource(s)

EBP 3. Determine which tool (that is, Word or FrameMaker) and which document template and style guide to use to create or update the document

EBP 4. Prepare estimate of time required to deliver guide (includes time to develop an annotated outline[1], to generate the first draft, to review, to revise, to finalize, to print the master copy, to generate the PDF file, to update the style guide, to document project peculiarities).

EBP 5. Process estimate into a P/O requisition or quote and send for approval (to Documentation Manager).

EBP 6. Review and adjust P/O requisition or quote (if necessary) and sends through formal approval process

EBP 7. Create new directory structure within the Documentation archive environment.

EBP 8. Assign a unique ID code to the document (*see procedure DOC-4 for more details*).

EBP 9. Determine who should review the document, other than the Documentation Manager[2], the Product Management Manager and a Quality Assurance representative (who are always part of the approval process).

EBP 10. Based on estimated draft delivery date, send advance notice to reviewers to indicate the date on which they will receive first draft and the date on which the review must end in order for the project to be successful.

[redacted], update its content and edit as required, based on the appropriate template and style guide. (All problems logged via the TJ system or via email against the previous versions of the document must be taken into consideration).

[redacted], develop it, based on the appropriate template and style guide.

EBP 13. Print a single copy of the draft and forward to Documentation Manager[2] for review.

EBP 14. Review draft to ensure that it meets policy standards, perform copyediting activities, and subsequently approve it for wide-scale review.

EBP 15. Update the generic Document Release and Sign-off Sheet to include the document specifics, the appropriate reviewer names, as well as the review end date for each reviewer (*see procedure DOC-2 for more details*).

[redacted][3], print a single copy of the draft, a single copy of the Document Release and Sign-off Sheet, and a single copy of the Proofreading Guidelines (*see procedure DOC-3 for more details*) and:

EBP 17. Forward copies of the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to the Documentation Manager[2], who reviews the document and performs copyediting activities, then signs off on the Document Release and Sign-off Sheet and forwards the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to the next reviewer indicated on the list.

[redacted], print copies of the draft (as many as there are reviewers), the Document Release and Sign-off Sheet, and the Proofreading Guidelines and:

EBP 19. Distribute copies of the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to all reviewers indicated on the list.

[redacted], send advisory to all reviewers that review process has begun, indicate the date on which the review process ends for each person, and remind them that the reviewed copies must be returned to the Documentation Manager.

EBP 21. Document reviewers perform copyediting activities as well as a substantive edit (that is, examine the contents of the document for completeness and organization).

EBP 22. Once all reviewed copies have been received, evaluate comments and forward to the technical writer.

EBP 23. Revise draft based on review comments[5] and finalize.

EBP 24. Send company-wide advisory to broadcast document's availability.

EBP 25. Print a master copy of document and file it in the Documentation filing cabinet.

EBP 26. If it is determined that the document should be available in print format, print another master copy and forward to Marketing for reproduction purposes.

EBP 27. Archive reviewed copies of document in the Documentation filing cabinet, along with completed Document Release and Sign-off Sheets.

EBP 28. Create PDF version of document.

EBP 29. If the document is associated with EliteSeries, send copy of PDF file to Marketing to be included on the EliteSeries Documentation CD[4].

EBP 30. Send copy of PDF file to R&D Administrative Assistant to post on the Intranet.

EBP 31. Send company-wide advisory to indicate document's availability on the Intranet.

EBP 32. Update list of completed documents and publish on Intranet.

EBP 33. Copy all soft copies of document into appropriate Documentation archive environment.

EBP 34. Update style guide, if required.
EBP 35. Conduct project wrap-up session to evaluate and document project successes and mistakes, dependencies, estimation accuracy, etc.
EBP 36. Archive project records and supporting materials.
Trigger ← Close job code.

1.2.5. Thread 1 - Documentation - User Doc. - Spanish Reference Guides

Note: The same process is used for both the English and Spanish production of the Reference Guides. The English process is detailed in the Application/Development section of the BPD, under Production of Reference Guides
Note: Reference Guides are scheduled to begin part way into the Spanish development phase, after enough work has been generated to port the help files to FrameMaker.

Trigger → Reference Guides are required, according to plan.
EBP 1. Create new directory structure in FrameMaker
EBP 2. Copy the master Reference Guide template in FM
EBP 3. Update the Lib Integration sheet (*see Appendix A*)
EBP 4. Update Preface, Front matter, Menu hierarchies with latest information or changes
EBP 5. Create a book file for every module (~ 20 books)
EBP 6. Format Preface
EBP 7. Modify and format Introduction (content, headers, footers)
EBP 8. Port for each book: Table of Contents, Introduction, User's Reference, Appendices (if applicable), Index, Feedback Sheet and What's New sections.
EBP 9. Port all options that are unchanged from previous release (*See Appendix A - Tracked via Translation's control document called 6.X.xls.*)
EBP 10. Port & format all new options from scratch (*See 6.1.xls: New? column*) [redacted], copy previous version of help file into current FM structure & incorporates diff into FM (*See 6.1.xls: Holding column*) [redacted], port & format all modified options from scratch into FM (*See 6.1.xls: Holding column*)
EBP 13. Update control document after option/library has been imported into FM
EBP 14. Ensure that the date in the Holding column of the control document is earlier than the date the file was imported into FM (i.e. ensure no changes have been made to the help file since the option was imported into FM).
EBP 15. Integrate libs
EBP 16. Integrate new report samples
EBP 17. Generate Book Lists, TOCs and Indices for all books
EBP 18. Edit TOCs and Indices. Regenerate, if necessary.
EBP 19. Print drafts of all books
EBP 20. QC layout of all books and edit if necessary.
EBP 21. Print final version of all books
EBP 22. Create PDF files for each book, to be posted on the Intranet
EBP 23. Create PDF links, to enable users to open documents from the Intranet
EBP 24. Test PDF links
EBP 25. Create READ.ME (notice of completion)

EBP 26. Post Reference Guides & READ.ME to the Intranet
EBP 27. Final QC of all links
EBP 28. Send email to Marketing indicating that the Reference Guides are ready to be added to the Documentation CD
Trigger ← Email to the company that Reference Guides are available on the Intranet

1.2.6. Thread 1 - Documentation - Technical Doc. - English Installation Guides

Trigger → Installation Guides are required, according to plan.
EBP 1. Finalize installation script (SE).
EBP 2. Open job code.
EBP 3. Allocate resource(s).
EBP 4. Determine which tool (that is, Word or FrameMaker) and which document template and style guide to use to create or update the document.
EBP 5. Prepare estimate of time required to deliver guide (includes time to develop an annotated outline[1], to generate the first draft, to review, to revise, to finalize, to print the master copy, to generate the PDF file, to update the style guide, to document project peculiarities).
EBP 6. Process estimate into a P/O requisition or quote and send to Documentation Manager for approval.
EBP 7. Review and adjust P/O requisition or quote (if necessary) and send through formal approval process
EBP 8. Create new directory structure within the Documentation archive environment.
EBP 9. Assign a unique identification code to the document (*see procedure DOC-4 for more details*).
EBP 10. Determine who should review the document, other than the Documentation Manager[2], the Product Management Manager, a Quality Assurance representative, and a Software Engineering representative.
EBP 11. Based on estimated draft delivery date, send advance notice to reviewers to indicate the date on which they will receive first draft and the date on which the review must end in order for the project to be successful.
[redacted], update its content and edit as required, based on the appropriate template and style guide. (All problems logged via the TJ system or via email against the previous versions of the document must be taken into consideration).
[redacted], develop it, based on the appropriate template and style guide.
EBP 14. Print a single copy of the draft and forward to Documentation Manager.
EBP 15. Review draft to ensure that it meets policy standards, perform copyediting activities, and subsequently approve it for wide-scale review.
EBP 16. Update the generic Document Release and Sign-off Sheet to include the document specifics, the appropriate reviewer names, as well as the review end date for each reviewer (*see procedure DOC-2 for more details*).
[redacted], print a single copy of the draft, a single copy of the Document Release and Sign-off Sheet, and a single copy of the Proofreading Guidelines (*see procedure DOC-3 for more details*) and:

EBP 18. Forward copies of the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to the Documentation Manager[2], who reviews the document and performs copyediting activities, then signs off on the Document Release and Sign-off Sheet and forwards the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to the next reviewer indicated on the list.

[redacted], print copies (as many as there are reviewers) of the draft, the Document Release and Sign-off Sheet, and the Proofreading Guidelines and:

EBP 20. Distribute copies of the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to all reviewers indicated on the list.

[redacted], send advisory to all reviewers that review process has begun, indicate the date on which the review process ends for each person, and remind them that the reviewed copies must be returned to the Documentation Manager.

EBP 22. Document reviewers perform copyediting activities as well as a substantive edit (that is, examine the contents of the document for completeness and organization).

EBP 23. Once all reviewed copies have been received, evaluate comments and forward to the technical writer.

EBP 24. Revise draft based on review comments[5] and finalize.

EBP 25. Send company-wide advisory to broadcast document's availability.

EBP 26. Print two master copies of document.

EBP 27. Forward one master copy to Marketing for reproduction purposes.

EBP 28. File one master copy in the Documentation filing cabinet.

EBP 29. Archive reviewed copies of document in the Documentation filing cabinet, along with completed Document Release and Sign-off Sheets.

EBP 30. Create PDF version of document.

EBP 31. If the document is associated with ***EliteSeries***, send copy of PDF file to Marketing to be included on the ***EliteSeries*** Documentation CD[4].

EBP 32. Send copy of PDF file to R&D Administrative Assistant to post on the Intranet.

EBP 33. Send company-wide advisory to indicate document's availability on the Intranet.

EBP 34. Update list of completed documents and publish on Intranet.

EBP 35. Copy all soft copies of document into appropriate Documentation archive environment.

EBP 36. Update style guide, if required.

EBP 37. Conduct project wrap-up session to evaluate and document project successes and mistakes, dependencies, estimation accuracy, etc.

EBP 38. Archive project records and supporting materials.

Trigger ← Close job code

1.2.7. Thread 1 - Documentation - Technical Doc. - English EliteSeries Documentation CD

Trigger → Documentation CD Guide is required, according to plan.

EBP 1. Open job code.

EBP 2. Allocate resource(s).

EBP 3. Prepare estimate of time required to deliver guide (includes time to develop an annotated outline[1], to generate the first draft, to review, to revise, to finalize, to print the master copy, to generate the PDF file, to update the style guide, to document project peculiarities).

EBP 4. Process estimate into a P/O requisition or quote and send to Documentation Manager for approval.

EBP 5. Review and adjust P/O requisition or quote (if necessary) and send through formal approval process.

EBP 6. Create new directory structure within the Documentation archive environment.

EBP 7. Assign a unique identification code to the document (*see procedure DOC-4 for more details*).

EBP 8. Based on estimated draft delivery date, send advance notice to reviewers to indicate the date on which they will receive first draft and the date on which the review must end in order for the project to be successful.

EBP 9. Since the document exists, update its content and edit as required, based on the appropriate template and style guide. (All problems logged via the TJ system or via email against the previous versions of the document must be taken into consideration).

EBP 10. Print a single copy of the draft and forward to Documentation Manager[2].

EBP 11. Review draft to ensure that it meets policy standards, perform copyediting activities, and subsequently approve it for wide-scale review.

EBP 12. Update the generic Document Release and Sign-off Sheet to include the document specifics, the appropriate reviewer names (that is, the Documentation Manager, the Product Management Manager and a Quality Assurance representative), as well as the review end date for each reviewer (*see procedure DOC-2 for more details*).

[redacted], print a single copy of the draft, a single copy of the Document Release and Sign-off Sheet, and a single copy of the Proofreading Guidelines (*see procedure DOC-3 for more details*) and:

EBP 14. Forward copies of the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to the Documentation Manager[2], who reviews the document and performs copyediting activities, then signs off on the Document Release and Sign-off Sheet and forwards the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to the next reviewer indicated on the list.

[redacted], print copies (as many as there are reviewers) of the draft, the Document Release and Sign-off Sheet, and the Proofreading Guidelines and:

EBP 16. Distribute copies of the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to all reviewers indicated on the list.

[redacted], send advisory to all reviewers that review process has begun, indicate the date on which the review process ends for each person, and remind them that the reviewed copies must be returned to the Documentation Manager.

EBP 18. Document reviewers perform copyediting activities as well as a substantive edit (that is, examine the contents of the document for completeness and organization).
EBP 19. Once all reviewed copies have been received, evaluate comments and forward to the technical writer.
EBP 20. Revise draft, based on review comments[5], and finalize.
EBP 21. Send company-wide advisory to broadcast document's availability.
EBP 22. Print a master copy of document and file it in the Documentation filing cabinet.
EBP 23. Print another master copy and forward to Marketing for reproduction purposes.
EBP 24. Archive reviewed copies of document in the Documentation filing cabinet, along with completed Document Release and Sign-off Sheets.
EBP 25. Create PDF version of document.
EBP 26. Send copy of PDF file to R&D Administrative Assistant to post on the Intranet.
EBP 27. Send company-wide advisory to indicate document's availability on the Intranet.
EBP 28. Update list of completed documents and publish on Intranet.
EBP 29. Copy all soft copies of document into appropriate Documentation archive environment.
EBP 30. Update style guide, if required.
EBP 31. Conduct project wrap-up session to evaluate and document project successes and mistakes, dependencies, estimation accuracy, etc.
EBP 32. Archive project records and supporting materials.
Trigger ← Close job code.

1.2.8. Thread 1 - Documentation - Technical Doc. - English Misc Guides

Trigger → A request is made to create/update and publish a new guide (ex. ***Elite****Series* Application Development Manual).
EBP 1. Open job code.
EBP 2. Allocate resource(s).
EBP 3. Determine which tool (that is, Word or FrameMaker) and which document template and style guide to use to create or update the document.
EBP 4. Prepare estimate of time required to deliver guide (includes time to develop an annotated outline[1], to generate the first draft, to review, to revise, to finalize, to print the master copy, to generate the PDF file, to update the style guide, to document project peculiarities).
EBP 5. Process estimate into a P/O requisition or quote and send to Documentation Manager for approval.
EBP 6. Review and adjust P/O requisition or quote (if necessary) and send through formal approval process.
EBP 7. Create new directory structure within the Documentation archive environment.
EBP 8. Assign a unique identification code to the document (*see procedure DOC-4 for more details*).
EBP 9. Determine who should review the document, other than the Documentation Manager[2], the Product Management Manager and a Quality Assurance representative (who are always part of the approval process).

EBP 10. Based on estimated draft delivery date, send advance notice to reviewers to indicate the date on which they will receive first draft and the date on which the review must end in order for the project to be successful.

[redacted], update its content and edit as required, based on the appropriate template and style guide. (All problems logged via the TJ system or via email against the previous versions of the document must be taken into consideration).

[redacted], develop it, based on the appropriate template and style guide.

EBP 13. Print a single copy of the draft and forward to Documentation Manager.

EBP 14. Review draft to ensure that it meets policy standards, perform copyediting activities, and subsequently approve it for wide-scale review.

EBP 15. Update the generic Document Release and Sign-off Sheet to include the document specifics, the appropriate reviewer names, as well as the review end date for each reviewer (*see procedure DOC-2 for more details*).

[redacted], print a single copy of the draft, a single copy of the Document Release and Sign-off Sheet, and a single copy of the Proofreading Guidelines (*see procedure DOC-3 for more details*) and:

EBP 17. Forward copies of the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to the Documentation Manager[2], who reviews the document and performs copyediting activities, then signs off on the Document Release and Sign-off Sheet and forwards the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to the next reviewer indicated on the list.

[redacted], print copies (as many as there are reviewers) of the draft, the Document Release and Sign-off Sheet, and the Proofreading Guidelines and:

EBP 19. Distribute copies of the draft, the Document Release and Sign-off Sheet and the Proofreading Guidelines to all reviewers indicated on the list.

[redacted], send advisory to all reviewers that review process has begun, indicate the date on which the review process ends for each person, and remind them that the reviewed copies must be returned to the Documentation Manager.

EBP 21. Document reviewers perform copyediting activities as well as a substantive edit (that is, examine the contents of the document for completeness and organization).

EBP 22. Once all reviewed copies have been received, evaluate comments and forward to the technical writer.

EBP 23. Revise draft based on review comments[5] and finalize.

EBP 24. Send company-wide advisory to broadcast document's availability.

EBP 25. Print a master copy of document and file it in the Documentation filing cabinet.

EBP 26. If it is determined that the document should be available in print format, print another master copy and forward to Marketing for reproduction purposes.

EBP 27. Archive reviewed copies of document in the Documentation filing cabinet, along with completed Document Release and Sign-off Sheets.

EBP 28. Create PDF version of document.

EBP 29. If the document is associated with ***EliteSeries***, send copy of PDF file to Marketing to be included on the ***EliteSeries*** Documentation CD[4].
EBP 30. Send copy of PDF file to R&D Administrative Assistant to post on the Intranet.
EBP 31. Send company-wide advisory to indicate document's availability on the Intranet.
EBP 32. Update list of completed documents and publish on Intranet.
EBP 33. Copy all soft copies of document into appropriate Documentation archive environment.
EBP 34. Update style guide, if required.
EBP 35. Conduct project wrap-up session to evaluate and document project successes and mistakes, dependencies, estimation accuracy, etc.
EBP 36. Archive project records and supporting materials.
Trigger ← Close job code.

1.2.9. Thread 1 - Documentation - Technical Doc. - English Data Model

Trigger → A request is made to print the data model
EBP 1. Print model on 8 1/2 x 14 paper
Trigger ← Delivery of model to person who requested it

[1] If the document does not currently exist and if it is predicted to be longer than 150 pages, then it is a good idea that an outline of the publication subsections be prepared for review prior to its full development. This will ensure that the writer's organizational strategy takes into account all relevant information about the product, audience, and tasks.

[2] If the Documentation Manager is not available to review the document and approve it for review, then an experienced senior resource (or Documentation Project Leader) must assume this responsibility. Either way, there must be one designated person available to ensure that EMD policies are respected throughout the TECSYS product documentation sets.

[3] If time permits, the **sequential review** is best. A single copy of the draft is printed and distributed to the first reviewer on the list, who reviews then distributes the draft to the next person on the list and so on and so forth. This process reduces the effort required by the author to update and finalize the document, as there is only one document to track and consult. On the other hand, a **parallel review** requires that multiple copies of the same document be distributed for review. As a result, review comments are often duplicated and the author must page through the various copies to find all of the changes requested (which is not very efficient). And, because reviewed copies are filed for future reference, this takes a considerable amount of physical space.

[4] This triggers the requirement to update the EliteSeries Documentation CD Guide.

[5] All review comments must be considered in the revision process. The enforcement of this policy is left to the author's integrity. The writer must initial each comment as the document is revised. In the case where a review note is not incorporated into the final document, the writer must initial the note and justify, in writing, the reason for its non-inclusion.

1.2.10. Thread 2 - Deployment - Prepare Environments - Front Office - Internal

The result of this thread is a final copy of the latest ***EliteSeries*** version on the Service machine for reference/shipping purposes, as well as one for CE, and a new starting point on HPElite for continued

development. Note that at this point, non PVCS environments are only for versions prior to 6.1, and for Spanish.

Trigger → QA completed
EBP 1. Disable auto-copy of released programs
EBP 2. Release all programs from the QA developer (also promote if in PVCS environment)
EBP 3. Run the integrity check script and verify results
EBP 4. Verify menu layout
EBP 5. Create security environment in directory
EBP 6. Execute LSPOSIM1 in new security environment (and add any new modules)
EBP 7. Update version number in base_rtn2.4gl
EBP 8. Update version number in utconnm3.4gl
EBP 9. Update version number in lslicem5.4gl
EBP 10. Update utconnm1.per (5.1) or u_version table (6.0 +) with ***EliteSeries*** version number
EBP 11. Execute LSMERGM1
EBP 12. Recompile UTCONNM1
EBP 13. Execute cpmenu to copy executables to QA
EBP 14. Unload the u_mod_sec table from security environment
EBP 15. Load the new u_mod_sec in QA live database
EBP 16. Load the new u_mod_sec in QA training database
EBP 17. Run relnew to create upgrade directory (tool not yet created for PVCS)
EBP 18. Remove executables and recompile QA environment
EBP 19. If in a non-PVCS environment:
    EBP 19. 1. Change QA pointer for release notes
    EBP 19. 2. Create master on HPElite
    EBP 19. 3. Create omega developer for new master on HPElite
EBP 20. If in a PVCS environment:
    EBP 20. 1. Apply version labels to frozen environment.
    EBP 20. 2. Create branches for further R&D development off of main trunk.
    EBP 20. 3. Create branch for maintenance releases off of main trunk.
EBP 21. Update Smaster variable in .ee files on HPElite
EBP 22. Create master on Service for shipping (see external preparation)
EBP 23. Create master on Service for CE
EBP 24. Generate new license for all masters just created
EBP 25. Enable auto-copy/compile of released programs
Trigger ← Environments ready for TECSYS teams

1.2.11. Thread 2 - Deployment - Prepare Environments - Front Office - External

The result of this thread is a final copy of the latest ***EliteSeries*** version on the Service machine for shipping purposes.

Trigger → Product in QA environment prepared/finalized

EBP 1. Create a new directory on the Service system that will be used to store the new release.

EBP 2. If in a non-PVCS environment:

- EBP 2. 1. From the QA environment, run *UTINSTM1* and create a new installation which includes the source code, live and training databases, la1, la2, and all modules plus the INSTALL and OBSOLETE modules.
- EBP 2. 2. Copy all files to the new directory.
- EBP 2. 3. Run the ***EliteSeries*** installation script.
- EBP 2. 4. Copy the upgrade directories (la1 and la2) from the QA environment to the new master environment.
- EBP 2. 5. The script file */qa/TOOLS/mkdoc* on Service must be updated to reflect the new version number

EBP 3. If in a PVCS environment:

- EBP 3. 1. From QA environment, run utinstm1 and create new installation which includes live and training databases, la1, la2, all modules including install module (do not include sources).
- EBP 3. 2. Copy all files to new directory.
- EBP 3. 3. Run Elite installation script.
- EBP 3. 4. Run mk_src to recreate sources from archives.
- EBP 3. 5. Run mk_new to generate upgrade directories.
- EBP 3. 6. Run utcompb1 to recompile all programs.

Trigger ← Environments ready for shipping

1.2.12. Thread 2 - Deployment - Prepare Environments - WMS

No processes currently exist for the preparation of WMS environments.

1.2.13. Thread 2 - Deployment - Prepare Environments - EliteQ & EliteView

No processes currently exist for the preparation of EliteQ and EliteView environments.

1.2.14. Thread 2 - Deployment - Shipping

The result of this thread is a copy of the latest ***EliteSeries*** version shipped to the client.

Note that currently the process in place to ensure that the issuance of this requisition is correct, is as follows:

- New shipments: Installation team reviews customer contract to ensure that the appropriate products are ordered, and that compatible versions are matched, and creates the requisition document from which the CDs are produced.
- Maint. Releases: Requisition document rolled forward from previous shipment, and verified by Installation team and Support, to ensure that the appropriate products are ordered, that compatible versions are matched, that the customer information is up to date. Requisition document then created from which the CDs are produced.

Trigger → Requisition document created from customer contract (includes machine, O/S, *EliteSeries* version, etc).

EBP 1. Open a session on the Service system as root.
EBP 2. Run *priv preptape and* choose the ***EliteSeries*** version from the menu.
EBP 3. Will now be brought into the Tape Creation Program.
EBP 4. Enter details as specified in requisition.
EBP 5. The program will create a file list and output file containing all requested material within the /TAPES/rel.6.0/tmp directory.
EBP 6. Using Windows Explorer, move file from the /TAPES/rel.6.0/tmp directory to the D:\ drive on the miscdwriter (*).
EBP 7. On the miscdwriter, start the HP-Sure program.
EBP 8. Click on the Explorer icon.
EBP 9. Copy the file from the D:\ drive to the HP-Sure window.
EBP 10. Click on the **write** (red) icon to create the CDROM.
EBP 11. CD sent to marketing for label printing and application.

Trigger ← Send tape or CD, shipping req. and tape req. to Admin for shipment to customer site.

Notes

- If shipping maintenance release due to contract agreement, only a packing slip is created. This is verified by Support and signed off by Install team manager (Ann).

- If shipping to a new customer, Elite bundled together with third party products, verified by installer against install checklist, and shipped via Fedex (no install guidelines or procedures included).

1.2.15. Thread 3 - Marketing - Product Features (New Customers)

EBP 1. Trigger → New product release and time available
EBP 2. Collect development analysis & design documents to scan for materials
EBP 3. Take list of new features for upgrade customers (when available) and update existing features document directly.
EBP 4. Provide materials to senior analysts for review
EBP 5. Marketing prepares new features document that integrates with existing features document.
EBP 6. Trigger ← New Customer product features complete

1.2.16. Thread 3 - Marketing - New Features (Upgrades)

EBP 1. Trigger → New product release and time available
EBP 2. Review release notes for new features
EBP 3. Collect development analysis & design documents to scan for materials
EBP 7. Provide materials to senior analysts for review
EBP 4. Marketing prepares new features
EBP 5. Trigger ← New features for existing customers complete

1.2.17. Thread 3 - Marketing – Product Profile

EBP 1. Trigger → New product release
EBP 2. Obtain product profile template from Marketing
EBP 3. Establish pricing (see 1.2.20)
EBP 4. Fill in information including: product name, version, languages supported, description, media availability, documentation, certification, software requirements, hardware requirements, list pricing, maintenance pricing, training and comments, ordering information (administration only)
EBP 5. Review content with marketing and make adjustments
EBP 6. Marketing posts contents to Marketing intranet and emails to sales organization
EBP 7. Trigger ← product profile complete

1.2.18. Thread 3 - Marketing - Press Releases

EBP 1. Trigger → Significant product event
EBP 2. Collect details on event (product release, customer implementation, etc)
EBP 3. Review press release prepared by marketing and note changes
EBP 4. Provide feedback to marketing
EBP 5. Press release is sent
EBP 6. Trigger ← Press release issued

1.2.19. Thread 3 - Marketing - Packaging

EBP 1. Trigger → New product nearing delivery/demonstration
EBP 2. Meet to review naming, versioning & coordination with other products
EBP 3. Meet to review application icons, splash screen, about window
EBP 4. Meet to review CD, installation process
EBP 5. Trigger ← Packaging complete

1.2.20. Thread 3 - Marketing - Pricing

EBP 1. Trigger → New product nearing delivery/demonstration
EBP 2. Provide input into pricing discussions regarding competitive/comparable product pricing
EBP 3. No active role in actual pricing process
EBP 4. Obtain list pricing information from marketing/sales
EBP 5. Trigger ← Pricing complete

1.2.21. Thread 4 - Training - Demo DB

No current process in place.

1.2.22. Thread 4 - Training - Course Material - Technical

No current process in place.
List of courses currently offered:

- Executive Information System for Administrators (1 day)

- Query Tool for Administrators (1 day)
- Query Tool for Developers

1.2.23. Thread 4 - Training - Course Material - User

No current process in place.
List of courses currently offered:

- Basic Software Functions for the Office (5 days)
- Complete Software Functions for the Office (10 days)
- Executive Information System for Users (1 day)
- Query Tool for Users - EliteQ (1 day)
- EDI (2 days)
- Warehouse Familiarization (1/2 day)
- Basic Software Functions for the Warehouse (2 days)
- Complete Software Functions for the Warehouse (5 days)

1.2.24. Thread 4 - Training - Course Material - Deployment

No current process in place.

1.2.25. Thread 4 - Training - Class (Teaching)

No current process in place.

1.2.26. Thread 5 - Spanish - Translation - Help

Trigger → Spanish version is required, according to plan. (Current arrangement is even numbered maintenance releases and feature releases.)

EBP 1. Set up scripts and structure of directories etc.
EBP 2. Run diff_fh (compare script) to determine scope of changes
EBP 3. Execute spdvlp on all programs, triggering an automatic update of the 4gl code for report type (r1, j1, b1, f1) programs.
EBP 4. Set project plan according to results of diff_fh and any other scheduled projects
EBP 5. Quote is approved. (Execute time_est script to determine how much time is needed for development.)
[redacted]. execute convert_help script to create a text file
EBP 7. Update control.doc in UNIX
EBP 8. Import English .doc file into Spanish Word template
EBP 9. Translate all components of the option
[redacted], execute osetup script, triggering set-up of Word document structure
EBP 11. Generate a Codewright diff to see what changes have been made
EBP 12. Execute overlay script
EBP 13. Translate new/modified information
EBP 14. Apply Spanish documentation bug fixes (D8's), if necessary
[redacted]. export (from Word into the development environment) and compile
EBP 16. Update 6.1.xls (Translation's control document. See Appendix A.)
EBP 17. Email DTP saying that file is ready for QC

EBP 18. Update control.doc in UNIX
EBP 19. QC, compile, test functionality. Make fixes if necessary and repeat process.
EBP 20. Release
EBP 21. Create diff and put it in the Holding directory (existing options only)
EBP 22. Update control document
Trigger ← Manager notifies SE saying that development is complete.

1.2.27. Thread 5 - Spanish - Deployment - Shipping

Trigger → Upgrade is requested, as per customer contract and preset delivery schedule.

Note: A Tape Req. is created from the information found within the customer contract (includes machine, O/S, EliteSeries version, etc). This information is updated periodically but there is currently no process in place to ensure that the information contained therein is correct.

Ann and Aline are working on this, however.

EBP 1. Open a session on the Service system
EBP 2. Run *priv preptape* and choose the appropriate ***EliteSeries*** version from the menu
EBP 3. At the main menu, run UTINSTM1
  EBP 3. 1. Enter the install name (usually an arbitrary number) and the description
  EBP 3. 2. Enter the details of the install, as specified in the tape req.
  EBP 3. 3. Enter the directory path and indicate which databases will be included
  EBP 3. 4. Enter the modules to be included, as specified in the tape req.
  EBP 3. 5. Create a list file and specify the command to create the tape or file
EBP 4. The program will create a list file.
EBP 5. Once created, the program will prompt the user to enter a tape before completing the procedure.
  [redacted] creating a tape, enter the tape and press Return
  [redacted], press Return if creating a file (location previously specified)
EBP 6. Once the process is complete:
  EBP 6. 1. Manually create a label for the tape to complete the cycle
  EBP 6. 2. Begin the process of creating the CD from the file, as specified below
EBP 7. Using FTP, move file from the STECSYS/tmp directory to the D:\ drive on the miscdwriter (*)
EBP 8. On the miscdwriter, start the HP-Sure program
EBP 9. Click on the Explorer icon
EBP 10. Copy the file from the D:\ drive to the HP-Sure window
EBP 11. Click on the **write** (red) icon to create the CDROM
EBP 12. Send request to Marketing for label to be printed
EBP 13. Label is ready (tape) or is received (CD) from Marketing
EBP 14. Apply label to tape or CD

Trigger ← Send tape or CD. shipping req. and tape req. to Admin for shipment to customer site

2. BUSINESS RULES

Business Rules define the rules that govern the process.

Business Rules ...

	Business Rule	Enforced	Effective
1.	Spanish translation: Update Translation's control doc (6.1.xls).	Y	H
2.	Spanish: Development environment must be set-up prior to start of translation.	N	H
3.	Spanish: Bugs (D8) must be properly logged and tracked for the purpose of fixing, not performance.	Y	H
4.	Spanish: Ownership must be defined and respected via control.doc file in UNIX.	Y	H
5.	Spanish: Delivery is 3-6 months after Eng/Fr. (Effectiveness not measured. This is a contractual obligation.)	N	N/A
6.	Spanish: Development done for even-numbered maintenance releases and all feature release. (Effectiveness not measured. This is a contractual obligation.)	Y	N/A
7.	Spanish: A job code must be created per release.	Y	H
8.	Spanish: An estimate per release must be prepared and approved.	Y	H
9.	Documentation: A job code must be created against which time is to be recorded.	Y	H
10.	Documentation: An estimate must be prepared.	Y	H
11.	Documentation: A P/O requisition or a quote must be approved by the Documentation Manager.	Y	H
12.	Documentation: A single repository for all documentation must exist.	Y	M
13.	Documentation: Time must be logged, for future analysis purposes.	N	H
14.	Documentation: If the document type is new, and an appropriate document template does not exist, then one must be created, along with an accompanying style guide. This must be done prior to the start of the project. If the document type exists already, then the appropriate template and style guides must be used.	Prev, Y. Now, N.	H
15.	Documentation: The document must be approved by the Documentation Manager or an experienced senior resource prior to being distributed for review.	Prev, Y. Now, N.	H
16.	Documentation: Document reviewers must receive advance notice that they will be called upon to review a given document. This enables them to plan their time accordingly.	Y	M
17.	Documentation: The document must be reviewed by technically competent resources and signed-off and approved for release to the user base.	Y	H
18.	Documentation: The technical writer must initial each review comment that he/she has included	Y	H

	and justify the non-inclusion of a given comment, if required, in writing.		
19.	Documentation: The document must be converted to the PDF file format for distribution.	Prev, Y. Now, ?.	H
20.	Documentation: All reviewed documents must be archived for future reference, along with completed Document Approval and Sign-off Sheets.	Prev, Y. Now, ?.	H
21.	Deployment: No deployment work may begin until QA completes their tests.	Y	H
22.	Deployment: Auto night compile turned off during this process.	Y	H
23.	Deployment: CDs always created from requisition documents.	Y	H
24.	Deployment: Internal master copy created on Service for Shipping.	Y	H
25.	Deployment: New development environments created for CE and RM.	Y	H
26.	Deployment: CDs must be shipped with documentation and Marketing package.	N	L
27.	Deployment: Admin takes care of actual shipping.		
28.	Training - A training schedule must be prepared.	Y	L
29.	Training - Manuals are to be published on the Intranet.	N	?
30.	Training - Manuals are to be updated with every new feature release.	N	?
31.	Marketing – Product profiles prepared prior to product release	N	Y
32.			
33.			
34.			

3. PERFORMANCE FACTORS

Performance Factors define how the process is to be measured in specific terms, and captures whether the measurement is actually tracked in today's operations, along with it's value.

Performance Measures ...

#	Category	Performance Description	Measurement	Track	Value
1.	Quality	Spanish: Help is only as good as the English.	??	??	??
2.	Quality	Spanish: Delivery, content, grammar, good business terms. The constraint is that the content is limited by the content of what you are using to translate from (English help).	??	??	??
3.	Customer Satisfaction	Documentation: Customer satisfaction has not been measured recently. How do we know that what we are delivering is valuable, useful and needed?	Not currently being measured.	N	H
4.	Cost	Documentation: Estimation accuracy	Effort vs Estimate in Hours	Y	H
5.	Efficiency/Timeliness	Documentation: Planning/Scheduling accuracy	Completion Date vs Planned Completion Date	Y	M
6.	Efficiency/Timeliness	Documentation: Document Review / Approval Period - Scheduling conflicts	Delay time between hand-offs and feedback	Y	M
7.	Quality	Documentation: Thoroughness of final document in terms of ensuring all components have been completed	Compare final document to annotated outline	N	M
8.	Quality	Documentation: Thoroughness of document templates and style guides	Compare final document against template	N	H
9.	Quality	Documentation: Document Review and approval Process	Completed Document Approval and Sign-off Sheets	Y	H
10.	Quality	Documentation: Thoroughness of document revision	Technical writer must initial each review comment, as each is addressed. If a comment is ignored, then	Y	H

			technical writer must justify in writing.		
11.	Training - Manual	Can be used as reference once class is over.			
12.	Training - Manual	Can be used generally irrespective of having attended class.			
13.	Deployment Quality	Documenting the CD creation process as it applies to the different formats. Readability - CDs. (Only applies to morons).	Not currently being measured	N	M
14.	Deployment Quality	Lead time and content verification - Is the right stuff being shipped?	Not currently being measured	N	H
15.	Deployment Quality	Ease-of-use of actual installation scripts, etc. Important because it's the first view of EliteSeries that our customers have.	Not currently being measured	N	H
16.	Marketing - Quality	Accurate communication of product features and capabilities	Not currently being measured	N	H
17.	Marketing - Timeliness	Delay between product release and availability of product information to the field	Not currently being measured	N	M
18.	Marketing - Impact	Impact of press releases on Stock price	Not currently being measured	N	M

4. TOOLS

The purpose of the Tool table is to define the various tools and or systems that are required to facilitate the process.

Tools ...

#	Tool	Purpose / Use	Comments	Value
1.	Intranet	Marketing	Sites not kept up-to-date/Needs to be reorganized by product.	H
2.	Spanish: MS Word - Help templates	To ensure a standard look & feel of all help, regardless of language	Essential. Works well but there are a few minor bugs in the conversion process back to UNIX (French accented characters).	H
3.	Spanish: Control doc. template in Excel	To ensure that all activities are tracked efficiently.	Essential. Works well when updated properly. Depends upon resource being conscientious.	H
4.	Spanish: FrameMaker and established templates within FM	To ensure standard look and feel of Reference Guides, regardless of language.	Works well but can be a bit buggy.	H
5.	Spanish: Acrobat Reader	To ensure proper viewing, printing, and searching functions can be effected/tested when creating links for Reference Guides via the Intranet.	Works well.	H
6.	Spanish: Telephone or person-to- person	To communicate urgent messages.	Works, except that not all in group have a phone. Follow up major decisions with written confirmation.	H
7.	Spanish: Email	To track work or communicate information	Efficient but does not work very well. Too many miscommunications. Replaces talking face-to-face too often.	H
8.	Spanish: Time Job	To ensure proper documenting of time on a specific job.	Works but is outdated.	H
9.	Spanish: .i	To view converted Word files in UNIX	Very useful for viewing MS Word converted results. Gives a good idea of what help screens will look like. A bit	H

			buggy.	
10.	Spanish: control.doc	Tracks ownership of an option	Works well. There is no overlap in development because it's clear who is working on the option at any given point in time.	H
11.	Spanish: Codewright	Aids in doing diffs, greps.	Is useful but the results are not always as concise as we would like. The tool is also not powerful. Example: You can't diff multiple apers at once.	H
12.	Spanish: SE's scripts	Automates some of our procedures	System needs an overhaul. There are too many scripts and we aren't sure what they are all used for. We are too dependent upon SE. More automation needed.	H
13.	Spanish: Dictionaries	Used to define, clarify business terms or grammar.	Works well.	M
14.	Spanish: Printer	Used to print the Reference Guide drafts and masters, as well as various other documents.	Works well.	H
15.	Documentation: MS Word established templates and style guides	To ensure a standard look and feel throughout a documentation set.	Essential, since others within the company (in addition to HSE and "CDI") are using this word processor. It is not, however, appropriate for documentation.	H
16.	Documentation: FrameMaker established templates and style guides	To ensure a standard look and feel throughout a documentation set and across multiple documentation sets.	Essential. Works well, but is in conflict with MS Word documents.	H
17.	Documentation: Acrobat PDF Writer	To convert documentation files to PDF format for wide distribution.	Essential. Works well.	H
18.	Documentation: Time Job	To ensure proper documenting of time on a specific job.	Essential. Works but is outdated.	H
19.	Documentation: Email	To send advisories of upcoming reviews and to broadcast document availability.	Essential. Works well.	H
20.	Documentation: Intranet	To ensure that published documents	Essential. Works well when updated on	M

		are available throughout the company.	a timely basis. Documents should not be grouped by department, however. They should be grouped by product.	
21.	Deployment: lsposim1			H
22.	Deployment: lsmergm1			H
23.	Deployment: c4	Compile menus & security options.		H
24.	Deployment: SQL	Load & unload tables		H
25.	Deployment: relnew	Create upgrade directory	For non-PVCS environment only.	H
26.	Deployment: PVCS	To apply version labels & create new branches.		H
27.	Deployment: unix commands	File, directory, and script creation, update and copying.		H
28.	Deployment: utinstm1	Create installations according to given parameters.		H
29.	Deployment: mk_src	Create source directories from archives.	PVCS environments only.	H
30.	Deployment: mk_new	Create upgrade directories.	PVCS environments only.	H
31.	Deployment: preptape	Tape creation program.		H
32.	Deployment: HP-Sure	CD writer software.		H
33.	Training - Word	Prepare manual		H
34.	Training - Micrografx	Prepare manual content		H
35.	Training - Release Notes	Update manuals and courses		H
36.	Training - Environment?	Class training		H
37.	Training - Training DB?	Class training		H
38.	Training - PowerPoint	Convey concept or information		M
39.	Training - Evaluation Form	Provide feedback		L
40.	Training - Equipment	Class training	Projector, one client per student, a server, flipchart.	H
41.	Marketing MS Word	To prepare documents	Press releases, product features etc. Should remain in a universally acceptable format like Word as opposed to using Framemaker	H

5. CONSTRAINTS

The constraint table is used to identify the constraints that have been imposed on the process.

Constraints ...

#	Constraint	Impact
1.	Spanish: Time, time, time - Deadlines and delivery dates are unrealistic. Does not allow for any improvement in our product - just a release. Help needs enormous overhaul with better text. Depends upon how clean the English is.	H
2.	Spanish: Lack of resources - Translation needs more resources to work in depth with the help, perhaps a few releases ahead of current projects. Also need to be able to accommodate Spanish bug fixing, which currently isn't being done consistently.	H
3.	Spanish: Transfer of knowledge - product, user and technical.	M
4.	Spanish: Translation should be done concurrently with English and French. Otherwise, there is a lack of continuity and it's difficult to plan other releases that are moving forward because the same resources are involved. (Issues to be resolved: timing, lack of resources, pre-established commitments to customers, company expectations on when help should be ready vs. realistic deadlines.	H
5.	Spanish: Effort permitted is often limited due to predefined time/cost constraints - jeopardizing best possible solution	H
6.	Documentation: If look and feel was finalized at the design level, resources would be able to begin the documentation process earlier.	H
7.	Documentation: Publications development process does not run in parallel with Application Development process. Documentation team is always in catch-up mode and is therefore unable to improve current publications and processes due to a lack of time.	H
8.	Documentation: Lack of resources.	M
9.	Documentation: Effort permitted is often limited due to predefined time/cost constraints, thereby jeopardizing best possible solution.	M
10.	Documentation: Knowledge transfer. Documentation team is rarely advised of upcoming product and/or technical training opportunities and feature release kick-off sessions. Given that this is the team that produces the materials that support users once training has ended, it simply makes sense that its resources should always be invited to these sessions. Failing this, their ignorance is transferred into the materials they produce.	M
11.	Documentation: MS Word vs. FrameMaker. Documentation team has chosen FrameMaker as its development tool of choice. It is, without a doubt, superior to MS Word in terms of page layout capabilities and large multi-chapter document manipulation. Other TECSYS departments, Haushahn, and "CDI" use MS Word, all with different templates and standards. As a result, each documentation set has a different look and feel. I suppose	M

	that currently this is acceptable since we are in catch up mode and there is a lack of documentation resources; however, if we don't address the problem of document standardization shortly, the number of documents to standardize will multiply tenfold and the standards that are currently in place shall not be respected due to confusion (the latter scenario has already occurred recently).	
12.	Deployment: Must be done be SE (technical resources).	H
13.	Deployment: Preparation of environments cannot be done in parallel with other development processes.	L
14.	Deployment: Preparation of environments may not be done during regular hours.	L
15.	Deployment: Lack of disk space means that old versions have to be archived.	L
16.	Training: Limited training facilities.	H
17.	Training: Lack of training facilitators.	H
18.	Training: Product has to be complete before training can begin. Complete look and feel should be finalized at the design document phase to permit training at an earlier stage.	H
19.	Marketing: Limited resources available to these processes	H

6. POSITIVE ASPECTS

The following table is used to capture all the positive aspects of the current process, in terms of anything that is working well and should potentially remain unchanged.

What works well ...

#	Positive Aspect	Comments	Value
1	Spanish: Quality	Product goes through several levels, ensuring the best possible quality, given the time and resource constraints.	H
2	Spanish: Standardization	Look and feel is uniform throughout product, based on template use.	H
3	Spanish: Tracking bugs	Bugs are tracked efficiently in TJ.	H
4	Spanish: Documented procedures	Procedures are well documented, making it easier to trace breakdowns and fix them efficiently. Also helpful to new resources who need to learn the procedures.	H
5	Spanish: Time estimates	Translation has a good idea of how much time is required to translate new programs due to accurate script (time_est).	H
6	Documentation: Documentation CD	Works well. Process is in place.	L
7	Documentation: Quality	Documents go through several editing levels as well as an established document approval process, ensuring the best possible quality, given the time and resource constraints.	H
8	Documentation: Standardization	Look and feel is uniform throughout a single documentation set, based on experienced technical writers.	M
9	Documentation: Page Layout Tool	The standard page layout program (FrameMaker) used by the Documentation group is efficient, powerful and appropriate.	M
10	Deployment: utinstm1	Is a dynamic tool, allowing user to specify the criteria for a required install, and stages/prepares all the appropriate files together, ready to be written to tape or CD.	H
11	Deployment: CD creation	Tool is being created which allow non-technical resources to create CDs. What is the status of this? Will this project be dropped? More analysis required.	M
12	Marketing	Product information for EliteSeries 6.X much improved over previous releases	M

7. NEGATIVE ASPECTS

The following table is used to capture all the negative aspects of the current process, in terms of anything that is not working well and should reviewed for potential changes. This table is not used to capture ideas to improve the problem, rather, it is used to simply identify the issue.

What doesn't work well ...

#	Negative Aspect	Comments	Impact
1	Intranet	Current documents are kept there, but product sites should house all information related to one product. Not many people use the Intranet as a source of information. The process needs to be reviewed.	M
2	Spanish: Not part of Application/Development phase	Doing Spanish after the fact means time/resources/money are wasted duplicating some aspects of the work. There is no cost benefit.	H
3	Spanish: Scripts	They are unfriendly and there is a lack of documentation which explains where to find them, what they are used for and which variables need modifying before development can begin on a new release. What they do is unclear. How to use them is unclear. There are too many scripts. There are too many manual processes. **Consequence**: Setting up the environments is a long, complicated procedure. Translation is stuck and unable to move forward if SE is unavailable to change all the variables, etc.	H
4	Spanish: Timing	Spanish is an afterthought and time to produce is too short. Too much pressure to produce Spanish feature release while working on E/F maintenance releases or next E/F feature release.	H
5	Spanish: Bottleneck	When delays occur, Spanish gets stuck at QA/Rel phase. Result: Very little continuity and high frustration levels.	H
6	Spanish: Kick-off sessions, team meetings	Translation does not have a kick-off session before a feature release resulting in nobody being sure of which issues will be implemented for the current feature release, and which will be on hold until the next release. Not enough team meetings to iron out problems.	M

7	Spanish: Communication with Stratis	There is no one designated resource to handle Stratis' queries. Stratis is getting info from too many areas, causing miscommunications and frustration.	M
8	Spanish: Shipping	Currently, there are few processes in place to ship for Spanish releases, or partial releases.	H
9	Documentation: Data Model	Printing multiple copies of the data model. Now that we have a printer, whose responsibility is it to administer the requests for copies?	L
10	Documentation: Lack of a Manager	Currently, there is no manager and resources are reporting to different people. This means that there is little continuity.	H
11	Documentation	Current documentation is grouped by program, not by functionality. (Example: There are no User Guides. We don't have any documentation describing COOPs from top to bottom. This would be useful information!)	H
12	Documentation: Kick-off Sessions	Technical writers are rarely invited to feature release kick-off sessions, and therefore, are not getting the information that can only be acquired via such meetings.	M
13	Documentation: Standardization	Look and feel is not uniform across documentation sets. The problem is three-fold: (1) there is a lack of resources required to standardize documents, (2) a process has not been clearly defined for retrieving new information from documents updated by third-parties (e.g. Haushahn documents), and (3) a single resource has not been assigned the responsibility of monitoring the enforcement of Documentation standards.	H
14	Documentation: Tracking Problem Reports	Process is in place to track problems reported with various pieces of documentation, but people are either unaware of codes to use in TJ or simply not interested or disciplined in reporting the problems encountered.	H
15	Documentation: Documented Procedures	Many procedures are either not documented or not well documented. Due to a lack of resources, very little time is dedicated to this activity. Currently does not pose a serious problem as the Documentation team is in a catch-up phase. However, as the library of publications increases, such lack of procedures will create bottlenecks, duplication of effort and high frustration levels.	H
16	Documentation: Bottleneck	Delays occur when a reviewer is not available to look at a document. This generally impacts the time needed to finalize the document and/or the delivery date.	M
17	Documentation: Design documents not maintained	The Design documents are not being kept up-to-date. Information is often missing or inaccurate. This can be reflected in the resulting documentation.	H
18	Documentation: Process should run in parallel with	The Documentation team is always in catch-up mode and is therefore unable to improve publications or processes. Ultimately, products do not move forward.	H

	Application Development process		
19	Documentation: Estimates	Previous user guide development quotes were based on simple page-count metrics and the estimates were fairly accurate. However, due to a lack of history, estimating document maintenance will not be as straightforward. Accurate estimates will require analysis based on precedence.	M
20	Documentation: Templates and Style Guides	Standard document templates are not clearly identified and accompanying style guides are outlined but not well documented. The current team of technical writers has managed to maintain standards due to past experience only. New types of documents are currently emerging for which there is no agreed-upon consensus, with the result that these no longer conform with the existing publications. Unless the problem is addressed soon, the Documentation team will undoubtedly have to standardize a number of existing documents in a not-too-distant future. This problem will be aggravated if new resources are hired.	H
21	Deployment: Manual process	Preparation of environments can only be done by technical persons which understand the process thoroughly.	H
22	Deployment: Processes	No processes or scripts currently exist for the preparation of WMS, EliteQ, and EliteView environments.	M
23	Deployment: Processes	Too many manual processes interspersed with existing scripts.	M
24	Deployment: Documentation	Very little documentation exists in support of these processes. Can't easily be handed off to other resources.	H
25	Deployment - Communication	Lack of information re: third party product requirements (i.e. C Compiler)	H
26	Deployment - Communication	Requisition reflects the contract, not what was promised to the customer.	H
27	Deployment - Shipping	SE is not setup to create the necessary labels. Must depend on Marketing to do this. What happens when they are backed up? Some CDs go out with no labels!	H
28	Deployment - Shipping	Have a separate machine for shipping purposes. Why? Disk space constraints, security issues.	M
29	Training - Processes	No set or scheduled process. Internal/External training is very much a 'seat of the pants' process.	H
30	Training - Material	No clear ownership of course material. No clear ownership or accountability.	H
31	Training - Material	Course material is out of date and inadequate.	H
32	Training - Material	There is currently no course material offered in French or Spanish.	M
33	Training - Releases	No training for new releases. No proactive planning for product releases.	H

34	Training - Database	There is currently no training database. What is the status of the one currently being prepared?	H
35	Marketing	Product collateral information often put together in a rush	H
36	Marketing	Not enough information available on how to sell individual products	H
37	Marketing	Inconsistent packaging of product profiles	H
38	Marketing	No central "group independent" product intranet site to combine contents of all groups	M
39	Marketing	Press releases for products often released too early while product still in early stages. Resource limitations may significantly delay availability which causes confusion and frustration with customers and sales force.	H
40			

8. CRITICAL SUCCESS FACTORS

The following table represents those factors which are deemed to be absolutely critical to the success of the process and deliverables alike. In other words, if these factors are not applied, we are sure to fail.

Critical Success Factors ...

#	Critical Success Factor	Impact of NOT Implementing
1.	Release Notes	Information is missed?
2.	Spanish: Kick-off meetings	*Backtracking* once project has started to fix issues that weren't spelled out at the beginning of the project.
3.	Spanish: Time to deliver	*Quality* cannot be expected when time to deliver is unrealistic
4.	Spanish: Communication	*High standards* cannot be achieved and maintained if communication isn't there. Will affect deliverables.
5.	Spanish: Defining a feature release	*Waste of time and resources* if it's not clear what should comprise a feature release and what a maintenance release should contain.
6.	Spanish: Follow up on resources	*Unhappy* resources will not stay at TECSYS
7.	Spanish: Resource availability	*Neglect* of key areas which would improve our product. Product doesn't move forward.
8.	Spanish: Estimates/Quotes	*Delays* in deliverables. Creates frustration and bottlenecks once project has begun.
9.	Spanish: Shipping CDs	*Waste of time and resources.* Process is a convoluted mess at the moment.
10.	Spanish: Scripts	*Inability to move forward effectively.* Delays will cause deadlines to be

		pushed back and will cost TECSYS money.
11.	Spanish: Understanding the business issue, how functionality addresses a business task. It's important to get the business knowledge behind the functionality.	*Poor help. Dissatisfied customers.*
12.	Documentation: Good writing skills	Lower quality.
13.	Documentation: Product knowledge	Inaccurate, incomplete information.
14.	Documentation: Reviewer availability	Efficiency and cost: Continued delays due to availability constraints and potential costly rework.
15.	Documentation: Current document templates and style guides	A mish-mash of documents with different page layout properties within and across documentation sets. In other words, a convoluted mess.
16.	Documentation: Technically competent reviewers	Inaccurate, incomplete information.
17.	Documentation: Documented procedures	Efficiency and cost.
18.	Documentation: Tracking time	Time must be logged for future analysis purposes.
19.	Documentation: Single designated individual to ensure that Documentation policies are respected throughout the TECSYS product documentation sets.	A mish-mash of page layout and language styles within and across documentation sets.
20.	Documentation: The technical writer must initial each review comment that he/she has included and justify the non-inclusion of a given comment, if required, in writing.	Inaccurate, incomplete information.
21.	Deployment: Person(s) responsible to close out and create new environments must have thorough knowledge of development environments, release process, and be technically inclined.	Working on, and shipping incorrect product.
22.	Deployment: Completed in off hours.	Development people unable to work
23.	Deployment: Imperative that requisitions be correct.	Clients receive wrong product or version.
24.	Training - Understanding the audience expectations	Dissatisfied audience
25.	Training - Knowledge of product	Inaccurate/incomplete information
26.	Training - Good writing skills	Lower quality
27.	Training - Good communication skills	Lower quality
28.	Training - Appropriate class room environment	Lower quality
29.	Training - Equipment in good working condition	Lower quality

30.	Training - Published schedule	Missed opportunity
31.	Marketing - Effective product communication	Misconceptions
32.	Marketing - Up to date product information	Lost sales
33.		

9. APPENDIX A - EXPLANATION OF TERMS USED WITHIN DOCUMENTATION THREADS

OBJECTIVES: This document provides resources within Translation with guidelines on how to update the Translation control documents. Currently, there are two control documents that Translation uses to track its work. They contain information related to English help development, French and Spanish translation, Reference Guides for all languages and feature and maintenance releases. Each time a new feature release is introduced, a new Microsoft Excel document is created. The control documents must be updated regularly by all concerned.

The 6-0 and 6-1 control documents are similar in nature and have the same look and feel. The maintenance releases are sectioned by color to make it visually easier to track them.

File Location

- j:\EMD\Control Documents\6-0.xls
- j:\EMD\Control Documents\6-1.xls

9.1. CONTROL DOCUMENT WORKSHEETS

In part, the Translation control document contains the following worksheets:

- Main Sheet;
- Ref Guides;
- Lib Integration;

These worksheets are described in more detail below.

9.2. BREAKDOWN OF THE MAIN SHEET

The 6-0 Master Control Document contains all the components of the 6.0 feature release and its associated maintenance releases (e.g. 6.0.1, 6.0.2). Each maintenance release is color-coded.

The 6-1 Master Control Document contains all the components of the 6.1 feature release and its associated maintenance releases (e.g. 6.1.1, 6.1.2). Each maintenance release is color-coded.

The following sections contain the descriptions of each column. The first section holds the descriptions that are found in both control documents. The second section holds the descriptions that are found solely in the 6-0 control document. The third section holds the descriptions that are found solely in the 6-1 control document.

9.2.1. Shared Column Descriptions

- The **Doc ID** column contains a unique character, which identifies the application(s) to which the option code belongs. Refer to procedure DOC-4 (*Documentation Numbering and Control*) for more details. The **Doc ID** column is filled in the first time an option code appears and is only filled in once for each option code listed.
- The **Option Code** column contains the list of options belonging to a specific source module, in alphabetical order. Each time a resource works on an option code, a new row is inserted below the preceding one and the information is added there. Thus, each option code will have a varying number of lines. Refer to Step 5 of the *Update Procedures for the Main Sheet* for more details on how to insert blank lines in a worksheet.
- The **New?** column is filled in when an option code has been added for a feature release.
- The **Eng: Last Full QC** column contains a date which indicates when a full format and functional QC were last performed. This will generally occur shortly before the release date for a particular version.
- The **Eng Edited** column is updated whenever an English component of a particular option is updated.
- The **Eng QC** column is updated after an English component of a particular option has been updated and QC'd in UNIX (either format, functional or both).
- The **Fr: Last Full QC** and **Fr QC** columns are updated in the same manner as their English counterparts (see above).
- The **Frn Trans** column is updated when a French component of a particular function has undergone translation.
- The **QA/Rel** column is updated when a final format and functional QC have been performed and the program has been verified and approved for released.
- The **Eng Frame** column is updated when the English Word file is ported to FrameMaker and reformatted for the purposes of creating the Reference Guides. Note that in the case of a lib, this column is updated when the lib has been integrated into the FrameMaker file(s).
- The **Frn Frame** column is updated when the French Word file is ported to FrameMaker and reformatted for the purposes of creating the Reference Guides. Note that in the case of a lib, this column is updated when the lib has been integrated into the FrameMaker file(s).
- The **Spa Frame** column is updated when the Spanish Word file is ported to FrameMaker and reformatted for the purposes of creating the Reference Guides. Note that in the case of a lib, this column is updated when the lib has been integrated into the FrameMaker file(s).
- The **Spa: Last Full QC** and **Spa QC** columns are updated in the same manner as their English counterparts (see above).
- The **Spa Trans** column is updated when a Spanish component of a particular option has undergone translation.
- The **Holding** column is updated when changes occur to a Word file for a given language. A diff file is created for each language that has been affected by the change. The diff file is then moved to the English, French or Spanish holding directory.
- The **Overlay** column is updated after the overlay script has been run for a particular option.

- The **Release** column is updated when an option has been fully QC'd and released, following the overlay procedure.

9.2.2. 6-1 Column Descriptions

- As of October 1997, the **Faber** column is no longer active.
- The **Diction** column is updated when the diction table has been modified for a given option. It serves to identify options where new reports are required for a reference guide or report digest.
- The **Rpt Gen** column is updated when a new report is generated for a particular release. A report is generated for the purposes of inserting it into a reference guide or report digest. The requirement for a new report is triggered by the presence of an entry in the **Diction** column.
- The **Rpt Ins** column is updated when a new report is inserted into a reference guide or report digest. The insertion of a new report is triggered by the presence of an entry in the **Rpt Gen** column.

9.2.3. Update Procedures for the Main Sheet

1. Open Microsoft Excel and go to the location of the worksheet (see *File Location* above).

Hint: Once you've opened the control documents, they will be listed under the **File** option of the menu. It is faster to choose the appropriate control document from the list when you need to open it again.

2. The Main Sheet of both control documents is split into four sections. The top two sections show the headings for the columns. The lower left section always shows the option codes and the lower right section is used to update information.
3. To update information, find the option code in question by placing the cursor in the bottom left section. Press **Ctrl+F** and enter the option code.
4. If the option code only has one line of information, determine whether or not this line already contains data. Disregard the **New?** and **Eng: Last Full QC** columns when doing so. If there is no data, proceed to Step 4.1. Otherwise, go to Step 5.
   - 4.1 If the row does not contain data, enter your initials and the date there. Due to space considerations, it is important to maintain the standard format when updating a cell, that is: *XX Mmmdd yy*. [Initials (two uppercase letters), a space, Date (first three letters of the month, beginning with a capital), Day (always two numbers with no space before them), a space and Year (last two numbers only).] Example: PS Dec02 97
   - 4.2 If you make a mistake and need to fix the information in the cell, go to the cell and press **F2**, or place your cursor in the cell and click on the white editing field above the worksheet.
5. If the option code row already has information in it or there is more than one line of information, you need to insert a new blank line to enter your data. To do this, find the last line for the option code and place your cursor below it, on the gray numbered box on the far left of the worksheet. Click the left mouse button. This will highlight the entire row. Click the right mouse button and select **Insert**. This will insert an entire blank row below the current line. Copy the option code from the cell above and insert it into the blank row. Proceed to update the necessary columns.

6. Press **Ctrl+S** to save your changes. To exit the document, choose **Close** from the **File** menu.

9.3. BREAKDOWN OF THE REF GUIDES SHEET

This worksheet contains the dates of when each section of the Reference Guides was updated, for a given language and release. Each filename begins with a FrameMaker extension (**.fm**). For filenames with two components, the second component is named for a main section within the file. For example, in ***TOC.fm**, there are three main sections which need to be tracked in the control document. They are the **Copyrights & Trademarks**, the **Preface** and the **Table of Contents**. Each column in the worksheet is further divided into four sub-sections. Following is an explanation of the columns and sub-sections.

To update the worksheet, follow Steps 4.1 and 4.2 from the section entitled *Update Procedures for the Main Sheet*. Press **Ctrl+S** to save your changes.

9.3.1. Columns

- The ***TOC.fm: Copyrights & Trademarks** column tracks what work has been done on the Copyrights and Trademarks section.
- The ***TOC.fm: Preface** column tracks what work has been done on the Preface section.
- The ***TOC.fm: Table of Contents** column tracks what work has been done on the Table of Contents section.
- The **intro.fm: Generic Intro** column tracks what work has been done on the Introduction section.
- The **intro.fm: Menu Hierarchy** column tracks what work has been done on the Menu Hierarchy section.
- The **user_ref.fm** column tracks what work has been done in the User's Reference title page file.
- The **apx_**.fm** column tracks what work has been done in the Appendix file.
- The ***IX.fm** column tracks what work has been done in the Index file.
- The **feedback.fm** column tracks what work has been done on the Customer Feedback Sheet, which appears as the last section of every module.
- The ***.book** column has a different function from the other columns. The book file contains a list of all the FrameMaker files created for a given module. This file is used to link all components of the book together as well as to generate the table of contents and the index. The **Content Updated** sub-section is updated when the initial book list is finalized. The **QC'd** column is updated when the contents of the book list have been verified and approved for release.

9.3.2. Sub-sections

- The **Template Updated** column is filled in when the working template for an existing FrameMaker file has been updated for a new release.
- The **Content Updated** column is filled in when the contents of an existing FrameMaker file have been updated, either through editing or translation.
- The **Layout Updated** column is filled in when the layout of an existing FrameMaker file has been updated.

- The **QC'd** column is filled in when all changes to the three previous columns for a given FrameMaker file have been verified and approved for release.

9.4. BREAKDOWN OF THE LIB INTEGRATION SHEET

This worksheet is comprised of two main sections. The first section lists all libraries that are to be integrated into the FrameMaker files for reference guide purposes. The second section lists all libraries that are to be added to the Reference Guides as appendices. This worksheet is used for reference purposes only. Use the **Eng Frame**, **Spa Frame** and **Frn Frame** columns of the **Main Sheet** for tracking purposes.

9.4.1. Section 1: Integration of Libs for Reference Guide Purposes

- The **Lib Name** column contains the names of all libraries to be integrated into the FrameMaker files.
- The **Option Code** column contains the names of all options that use a particular library.
- The **Where to Integrate Sections** column lists detailed instructions on how to break down the library and integrate its components into a given FrameMaker file.

9.4.2. Section 2: Integration of Libs as Appendices for Reference Guide Purposes

- The **Lib Name** column contains the names of all libs to be integrated into FrameMaker as appendices.
- The **Module Name** column indicates into which reference guide the library must be placed.

ANNEXE 6

Rational Unified Process

Rational Unified Process:
Best Practices for Software Development Teams

Table of Contents

Abstract

This paper presents an overview of the Rational Unified Process.™ The Rational Unified Process is a software engineering process, delivered through a web-enabled, searchable knowledge base. The process enhances team productivity and delivers software best practices via guidelines, templates and tool mentors for all critical software lifecycle activities. The knowledge base allows development teams to gain the full benefits of the industry-standard Unified Modeling Language (UML).

What Is the Rational Unified Process?

The Rational Unified Process is a Software Engineering Process. It provides a disciplined approach to assigning tasks and responsibilities within a development organization. Its goal is to ensure the production of high-quality software that meets the needs of its end-users, within a predictable schedule and budget. [11, 13]

The Rational Unified Process is a process product, developed and maintained by Rational® Software. The development team for the Rational Unified Process are working closely with customers, partners, Rational's product groups as well as Rational's consultant organization, to ensure that the process is continuously updated and improved upon to reflect recent experiences and evolving and proven best practices.

The Rational Unified Process enhances team productivity, by providing every team member with easy access to a knowledge base with guidelines, templates and tool mentors for all critical development activities. By having all team members accessing the same knowledge base, no matter if you work with requirements, design, test, project management, or configuration management, we ensure that all team members share a common language, process and view of how to develop software.

The Rational Unified Process activities create and maintain models. Rather than focusing on the production of large amount of paper documents, the Unified Process emphasizes the development and maintenance of models--semantically rich representations of the software system under development. [3, 7, 8]

The Rational Unified Process is a guide for how to effectively use the Unified Modeling Language (UML). The UML is a industry-standard language that allows us to clearly communicate requirements, architectures and designs. The UML was originally created by Rational Software, and is now maintained by the standards organization Object Management Group (OMG). [4]

The Rational Unified Process is supported by tools, which automate large parts of the process. They are used to create and maintain the various artifacts-models in particular-of the software engineering process: visual modeling, programming, testing, etc. They are invaluable in supporting all the bookkeeping associated with the change management as well as the configuration management that accompanies each iteration.

The Rational Unified Process is a configurable process. No single process is suitable for all software development. The Unified Process fits small development teams as well as large development organizations. The Unified Process is founded on a simple and clear process architecture that provides commonality across a family of processes. Yet, it can be varied to accommodate different situations. It contains a Development Kit, providing support for configuring the process to suit the needs of a given organization.

The Rational Unified Process captures many of the best practices in modern software development in a form that is suitable for a wide range of projects and organizations. Deploying these best practices¾using the Rational Unified Process as your guide¾offers development teams a number of key advantages. In next section, we describe the six fundamental best practices of the Rational Unified Process.

Effective Deployment of 6 Best Practices

The Rational Unified Process describes how to effectively deploy commercially proven approaches to software development for software development teams. These are called "best practices" not so much because you can precisely quantify their value, but rather, because they are observed to be commonly used in industry by successful organizations. The Rational Unified Process provides each team member with the guidelines, templates and tool mentors necessary for the entire team to take full advantage of among others the following best practices:

1. Develop software iteratively
2. Manage requirements
3. Use component-based architectures
4. Visually model software
5. Verify software quality
6. Control changes to software

Develop Software Iteratively--Given today's sophisticated software systems, it is not possible to sequentially first define the entire problem, design the entire solution, build the software and then test the

product at the end. An iterative approach is required that allows an increasing understanding of the problem through successive refinements, and to incrementally grow an effective solution over multiple iterations. The Rational Unified Process supports an iterative approach to development that addresses the highest risk items at every stage in the lifecycle, significantly reducing a project's risk profile. This iterative approach helps you attack risk through demonstrable progress¾frequent, executable releases that enable continuous end user involvement and feedback. Because each iteration ends with an executable release, the development team stays focused on producing results, and frequent status checks help ensure that the project stays on schedule. An iterative approach also makes it easier to accommodate tactical changes in requirements, features or schedule. [1, 2, 10]

Manage Requirements--The Rational Unified Process describes how to elicit, organize, and document required functionality and constraints; track and document tradeoffs and decisions; and easily capture and communicate business requirements. The notions of use case and scenarios proscribed in the process has proven to be an excellent way to capture functional requirements and to ensure that these drive the design, implementation and testing of software, making it more likely that the final system fulfills the end user needs. They provide coherent and traceable threads through both the development and the delivered system. [7]

Use Component-Based Architectures--The process focuses on early development and baselining of a robust executable architecture, prior to committing resources for full-scale development. It describes how to design a resilient architecture that is flexible, accommodates change, is intuitively understandable, and promotes more effective software reuse. The Rational Unified Process supports component-based software development. Components are non-trivial modules, subsystems that fulfill a clear function. The Rational Unified Process provides a systematic approach to defining an architecture using new and existing components. These are assembled in a well-defined architecture, either ad hoc, or in a component infrastructure such as the Internet, CORBA, and COM, for which an industry of reusable components is emerging. [5]

Visually Model Software--The process shows you how to visually model software to capture the structure and behavior of architectures and components. This allows you to hide the details and write code using "graphical building blocks." Visual abstractions help you communicate different aspects of your software; see how the elements of the system fit together; make sure that the building blocks are consistent with your code; maintain consistency between a design and its implementation; and promote unambiguous communication. The industry-standard Unified Modeling Language (UML), created by Rational Software, is the foundation for successful visual modeling. [4, 12]

Verify Software Quality--Poor application performance and poor reliability are common factors which dramatically inhibit the acceptability of today's software applications. Hence, quality should be reviewed with respect to the requirements based on reliability, functionality, application performance and system performance. The Rational Unified Process assists you in the planning, design, implementation, execution, and evaluation of these test types. Quality assessment is built into the process, in all activities, involving all participants, using objective measurements and criteria, and not treated as an afterthought or a separate activity performed by a separate group.

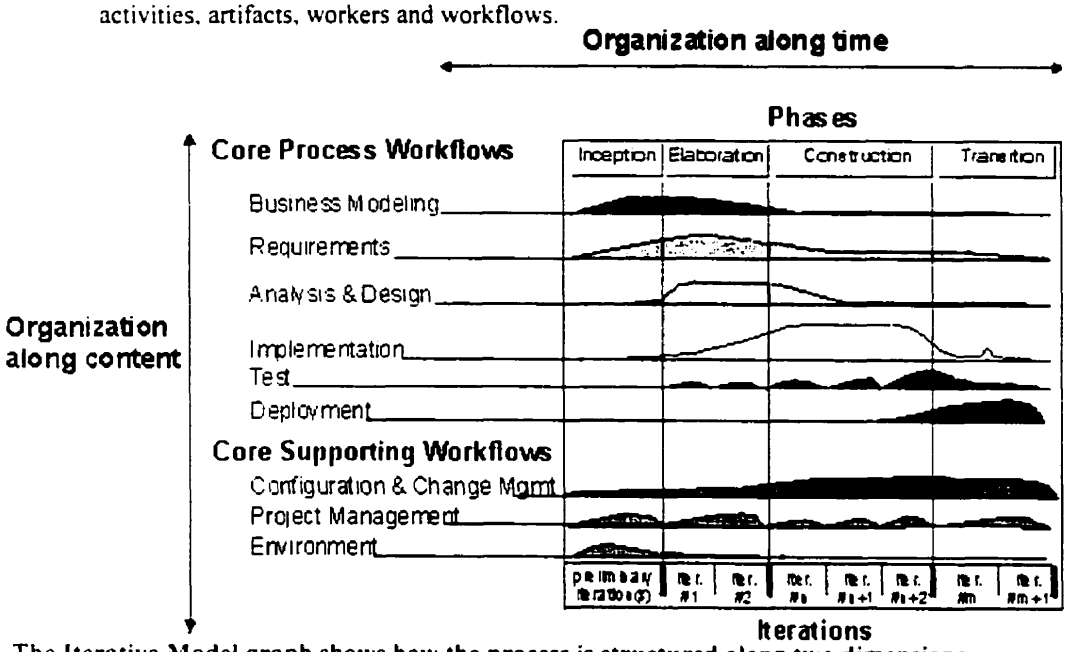
Control Changes to Software--The ability to manage change--making certain that each change is acceptable, and being able to track changes--is essential in an environment in which change is inevitable. The process describes how to control, track and monitor changes to enable successful iterative development. It also guides you in how to establish secure workspaces for each developer by providing isolation from changes made in other workspaces and by controlling changes of all software artifacts (e.g., models, code, documents, etc.). And it brings a team together to work as a single unit by describing how to automate integration and build management.

Process Overview

Two Dimensions

The process can be described in two dimensions, or along two axis:

- The horizontal axis represents time and shows the dynamic aspect of the process as it is enacted, and it is expressed in terms of cycles, phases, iterations, and milestones.
- The vertical axis represents the static aspect of the process: how it is described in terms of activities, artifacts, workers and workflows.



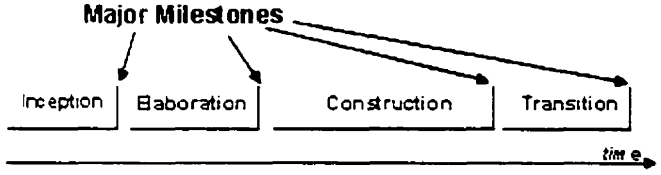
The Iterative Model graph shows how the process is structured along two dimensions.

Phases and Iterations--The Time Dimension

This is the dynamic organization of the process along time.
The software lifecycle is broken into cycles, each cycle working on a new generation of the product. The Rational Unified Process divides one development cycle in four consecutive phases. [10]

- Inception phase
- Elaboration phase
- Construction phase
- Transition phase

Each phase is concluded with a well-defined *milestone*--a point in time at which certain critical decisions must be made, and therefore key goals must have been achieved. [2]



The phases and major milestones in the process.

Each phase has a specific purpose.

Inception Phase

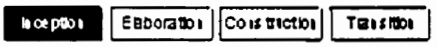
During the inception phase, you establish the business case for the system and delimit the project scope. To accomplish this you must identify all external entities with which the system will interact (actors) and

define the nature of this interaction at a high-level. This involves identifying all use cases and describing a few significant ones. The business case includes success criteria, risk assessment, and estimate of the resources needed, and a phase plan showing dates of major milestones. [10, 14]

The outcome of the inception phase is:

- A vision document: a general vision of the core project's requirements, key features, and main constraints.
- An initial use-case model (10%-20% complete).
- An initial project glossary (may optionally be partially expressed as a domain model).
- An initial business case, which includes business context, success criteria (revenue projection, market recognition, and so on), and financial forecast.
- An initial risk assessment.
- A project plan, showing phases and iterations.
- A business model, if necessary.
- One or several prototypes.

Milestone: Lifecycle Objectives



At the end of the inception phase is the first major project milestone: the Lifecycle Objectives Milestone. The evaluation criteria for the inception phase are:

- Stakeholder concurrence on scope definition and cost/schedule estimates.
- Requirements understanding as evidenced by the fidelity of the primary use cases.
- Credibility of the cost/schedule estimates, priorities, risks, and development process.
- Depth and breadth of any architectural prototype that was developed.
- Actual expenditures versus planned expenditures.

The project may be cancelled or considerably re-thought if it fails to pass this milestone.

Elaboration Phase

The purpose of the elaboration phase is to analyze the problem domain, establish a sound architectural foundation, develop the project plan, and eliminate the highest risk elements of the project. To accomplish these objectives, you must have the "mile wide and inch deep" view of the system. Architectural decisions have to be made with an understanding of the whole system: its scope, major functionality and nonfunctional requirements such as performance requirements.

It is easy to argue that the elaboration phase is the most critical of the four phases. At the end of this phase, the hard "engineering" is considered complete and the project undergoes its most important day of reckoning: the decision on whether or not to commit to the construction and transition phases. For most projects, this also corresponds to the transition from a mobile, light and nimble, low-risk operation to a high-cost, high-risk operation with substantial inertia. While the process must always accommodate changes, the elaboration phase activities ensure that the architecture, requirements and plans are stable enough, and the risks are sufficiently mitigated, so you can predictably determine the cost and schedule for the completion of the development. Conceptually, this level of fidelity would correspond to the level necessary for an organization to commit to a fixed-price construction phase.

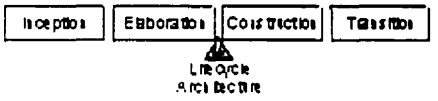
In the elaboration phase, an executable architecture prototype is built in one or more iterations, depending on the scope, size, risk, and novelty of the project. This effort should at least address the critical use cases identified in the inception phase, which typically expose the major technical risks of the project. While an

evolutionary prototype of a production-quality component is always the goal. this does not exclude the development of one or more exploratory, throw-away prototypes to mitigate specific risks such as design/requirements trade-offs, component feasibility study, or demonstrations to investors, customers, and end-users.

The outcome of the elaboration phase is:

- A use-case model (at least 80% complete) - all use cases and actors have been identified, and most use-case descriptions have been developed.
- Supplementary requirements capturing the non functional requirements and any requirements that are not associated with a specific use case.
- A Software Architecture Description.
- An executable architectural prototype.
- A revised risk list and a revised business case.
- A development plan for the overall project, including the coarse-grained project plan, showing iterations" and evaluation criteria for each iteration.
- An updated development case specifying the process to be used.
- A preliminary user manual (optional).

Milestone: Lifecycle Architecture



At the end of the elaboration phase is the second important project milestone, the Lifecycle Architecture Milestone. At this point, you examine the detailed system objectives and scope, the choice of architecture, and the resolution of the major risks.

The main evaluation criteria for the elaboration phase involves the answers to these questions:

- Is the vision of the product stable?
- Is the architecture stable?
- Does the executable demonstration show that the major risk elements have been addressed and credibly resolved?
- Is the plan for the construction phase sufficiently detailed and accurate? Is it backed up with a credible basis of estimates?
- Do all stakeholders agree that the current vision can be achieved if the current plan is executed to develop the complete system, in the context of the current architecture?
- Is the actual resource expenditure versus planned expenditure acceptable?

The project may be aborted or considerably re-thought if it fails to pass this milestone.

Construction Phase

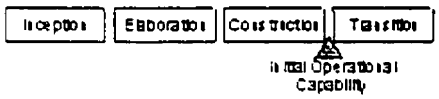
During the construction phase, all remaining components and application features are developed and integrated into the product, and all features are thoroughly tested. The construction phase is, in one sense, a manufacturing process where emphasis is placed on managing resources and controlling operations to optimize costs, schedules, and quality. In this sense, the management mindset undergoes a transition from the development of intellectual property during inception and elaboration, to the development of deployable products during construction and transition.

Many projects are large enough that parallel construction increments can be spawned. These parallel activities can significantly accelerate the availability of deployable releases; they can also increase the complexity of resource management and workflow synchronization. A robust architecture and an understandable plan are highly correlated. In other words, one of the critical qualities of the architecture is its ease of construction. This is one reason why the balanced development of the architecture and the plan is stressed during the elaboration phase.

The outcome of the construction phase is a product ready to put in hands of its end-users. At minimum, it consists of:

- The software product integrated on the adequate platforms.
- The user manuals.
- A description of the current release.

Milestone: Initial Operational Capability



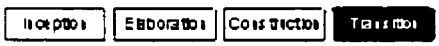
At the end of the construction phase is the third major project milestone (Initial Operational Capability Milestone). At this point, you decide if the software, the sites, and the users are ready to go operational, without exposing the project to high risks. This release is often called a "beta" release.

The evaluation criteria for the construction phase involve answering these questions:

- Is this product release stable and mature enough to be deployed in the user community?
- Are all stakeholders ready for the transition into the user community?
- Are the actual resource expenditures versus planned expenditures still acceptable?

Transition may have to be postponed by one release if the project fails to reach this milestone.

Transition Phase



The purpose of the transition phase is to transition the software product to the user community. Once the product has been given to the end user, issues usually arise that require you to develop new releases, correct some problems, or finish the features that were postponed.

The transition phase is entered when a baseline is mature enough to be deployed in the end-user domain. This typically requires that some usable subset of the system has been completed to an acceptable level of quality and that user documentation is available so that the transition to the user will provide positive results for all parties. This includes:

- "Beta testing" to validate the new system against user expectations
- Parallel operation with a legacy system that it is replacing
- Conversion of operational databases
- Training of users and maintainers
- Roll-out the product to the marketing, distribution, and sales teams

The transition phase focuses on the activities required to place the software into the hands of the users. Typically, this phase includes several iterations, including beta releases, general availability releases, as well as bug-fix and enhancement releases. Considerable effort is expended in developing user-oriented documentation, training users, supporting users in their initial product use, and reacting to user feedback.

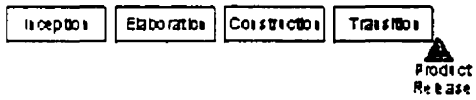
At this point in the lifecycle, however, user feedback should be confined primarily to product tuning, configuring, installation, and usability issues.

The primary objectives of the transition phase include:

- Achieving user self-supportability
- Achieving stakeholder concurrence that deployment baselines are complete and consistent with the evaluation criteria of the vision
- Achieving final product baseline as rapidly and cost effectively as practical

This phase can range from being very simple to extremely complex, depending on the type of product. For example, a new release of an existing desktop product may be very simple, whereas replacing a nation's air-traffic control system would be very complex.

Milestone: Product Release



At the end of the transition phase is the fourth important project milestone, the Product Release Milestone. At this point, you decide if the objectives were met, and if you should start another development cycle. In some cases, this milestone may coincide with the end of the inception phase for the next cycle.

The primary evaluation criteria for the transition phase involve the answers to these questions:

- Is the user satisfied?
- Are the actual resources expenditures versus planned expenditures still acceptable?

Iterations

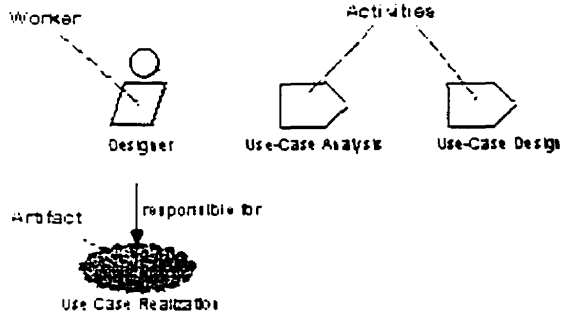
Each phase in the Rational Unified Process can be further broken down into iterations. An iteration is a complete development loop resulting in a release (internal or external) of an executable product, a subset of the final product under development, which grows incrementally from iteration to iteration to become the final system. [10]

Benefits of an Iterative Approach

Compared to the traditional waterfall process, the iterative process has the following advantages:

- Risks are mitigated earlier
- Change is more manageable
- Higher level of reuse
- The project team can learn along the way
- Better overall quality

Static Structure of the Process



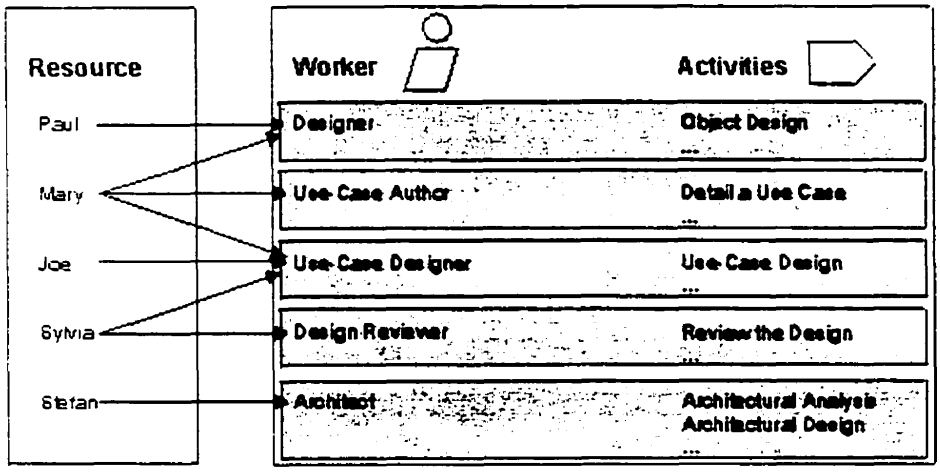
A process describes who is doing what, how, and when. The Rational Unified Process is represented using four primary modeling elements:

- Workers, the "who"
- Activities, the "how"
- Artifacts, the "what"
- Workflows, the "when"

Activities, Artifacts, and Workers

Workers, activities, and artifacts.
Worker

A *worker* defines the behavior and responsibilities of an individual, or a group of individuals working together as a team. You could regard a worker as a "hat" an individual can wear in the project. One individual may wear many different hats. This is an important distinction because it is natural to think of a worker as the individual or team itself, but in the Unified Process the worker is more the role defining how the individuals should carry out the work. The responsibilities we assign to a worker includes both to perform a certain set of activities as well as being owner of a set of artifacts.



People and workers.

Activity

An *activity* of a specific worker is a unit of work that an individual in that role may be asked to perform. The activity has a clear purpose, usually expressed in terms of creating or updating some artifacts, such as a model, a class, a plan. Every activity is assigned to a specific worker. The granularity of an activity is

generally a few hours to a few days, it usually involves one worker, and affects one or only a small number of artifacts. An activity should be usable as an element of planning and progress: if it is too small, it will be neglected, and if it is too large, progress would have to be expressed in terms of an activity's parts.

Example of activities:

- Plan an iteration, for the Worker: Project Manager
- Find use cases and actors, for the Worker: System Analyst
- Review the design, for the Worker: Design Reviewer
- Execute performance test, for the Worker: Performance Tester

Artifact

An *artifact* is a piece of information that is produced, modified, or used by a process. Artifacts are the tangible products of the project, the things the project produces or uses while working towards the final product. Artifacts are used as input by workers to perform an activity, and are the result or output of such activities. In object-oriented design terms, as activities are operations on an active object (the worker), artifacts are the parameters of these activities.

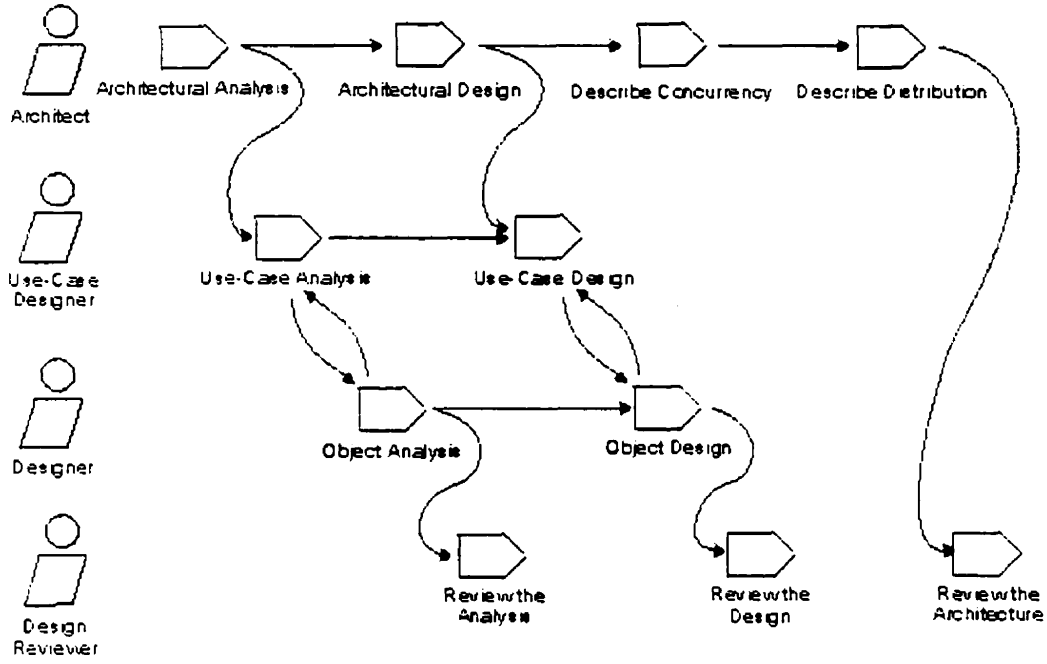
Artifacts may take various shapes or forms:

- A model, such as the Use-Case Model or the Design Model
- A model element, i.e. an element within a model, such as a class, a use case or a subsystem
- A document, such as Business Case or Software Architecture Document
- Source code
- Executables

Workflows

A mere enumeration of all workers, activities and artifacts does not quite constitute a process. We need a way to describe meaningful sequences of activities that produce some valuable result, and to show interactions between workers.

A *workflow* is a sequence of activities that produces a result of observable value.
In UML terms, a workflow can be expressed as a sequence diagram, a collaboration diagram, or an activity diagram. We use a form of activity diagrams in this white paper.

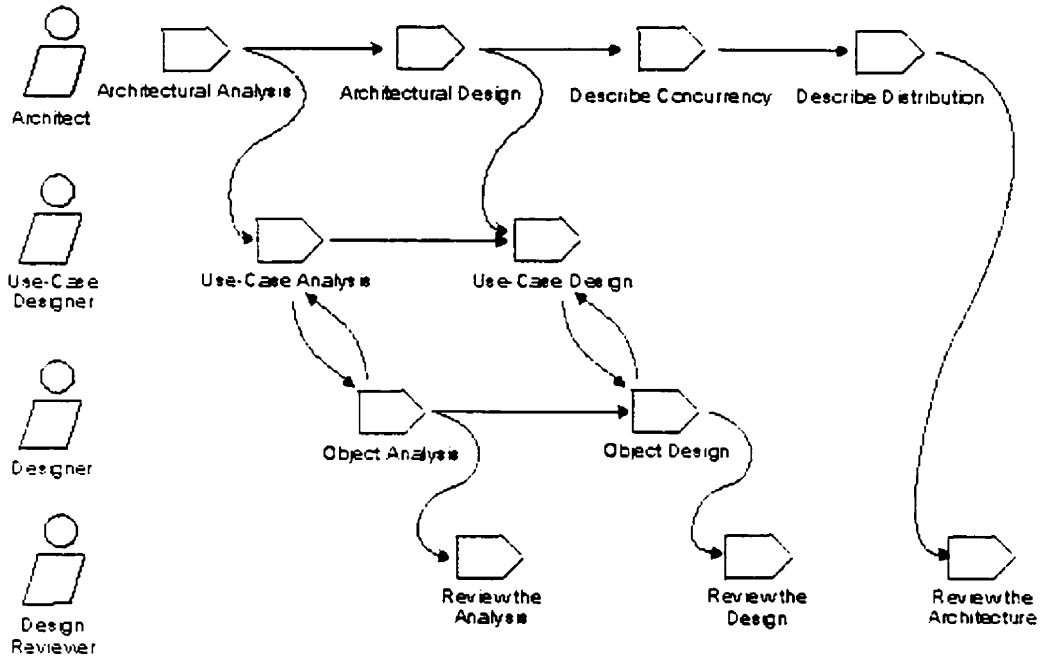


Example of workflow.

Note that it is not always possible or practical to represent all of the dependencies between activities. Often two activities are more tightly interwoven than shown. especially when they involve the same worker or the same individual. People are not machines, and the workflow cannot be interpreted literally as a program for people, to be followed exactly and mechanically. In next section we will discuss the most essential type of workflows in the process, called Core Workflows.

Core Workflows

There are nine core process workflows in the Rational Unified Process, which represent a partitioning of all workers and activities into logical groupings.



The nine core process workflows.

The core process workflows are divided into six core "engineering" workflows:

1. Business modeling workflow
2. Requirements workflow
3. Analysis & Design workflow
4. Implementation workflow
5. Test workflow
6. Deployment workflow

And three core "supporting" workflows:

1. Project Management workflow
2. Configuration and Change Management workflow
3. Environment workflow

Although the names of the six core engineering workflows may evoke the sequential phases in a traditional waterfall process, we should keep in mind that the phases of an iterative process are different and that these workflows are revisited again and again throughout the lifecycle. The actual complete workflow of a project interleaves these nine core workflows, and repeats them with various emphasis and intensity at each iteration.

Business Modeling

One of the major problems with most business engineering efforts, is that the software engineering and the business engineering community do not communicate properly with each other. This leads to that the output from business engineering is not used properly as input to the software development effort, and vice-versa. The Rational Unified Process addresses this by providing a common language and process for both communities, as well as showing how to create and maintain direct traceability between business and software models.

In Business Modeling we document business processes using so called business use cases. This assures a common understanding among all stakeholders of what business process needs to be supported in the

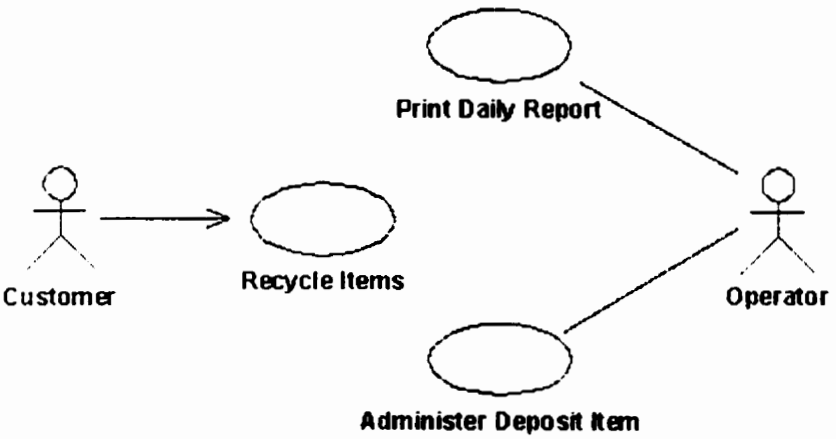
organization. The business use cases are analyzed to understand how the business should support the business processes. This is documented in a business object-model.

Many projects may choose not to do business modeling.

Requirements

The goal of the Requirements workflow is to describe what the system should do and allows the developers and the customer to agree on that description. To achieve this, we elicit, organize, and document required functionality and constraints; track and document tradeoffs and decisions.

A Vision document is created, and stakeholder needs are elicited. Actors are identified, representing the users, and any other system that may interact with the system being developed. Use cases are identified, representing the behavior of the system. Because use cases are developed according to the actor's needs, the system is more likely to be relevant to the users. The following figure shows an example of a use-case model for a recycling-machine system.



An example of a use-case model with actors and use cases.

Each use case is described in detail. The use-case description shows how the system interacts step by step with the actors and what the system does. Non-functional requirements are described in Supplementary Specifications.

The use cases function as a unifying thread throughout the system's development cycle. The same use-case model is used during requirements capture, analysis & design, and test.

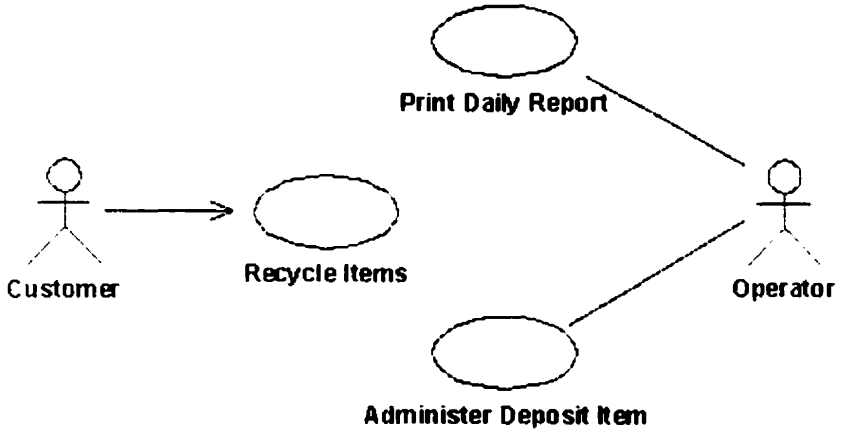
Analysis and Design

The goal of the Analysis and Design workflow is to show how the system will be realized in the implementation phase. You want to build a system that:

- Performs-in a specific implementation environment-the tasks and functions specified in the use-case descriptions.
- Fulfills all its requirements.
- Is structured to be robust (easy to change if and when its functional requirements change).

Analysis and Design results in a design model and optionally an analysis model. The design model serves as an abstraction of the source code; that is, the design model acts as a 'blueprint' of how the source code is structured and written.

The design model consists of design classes structured into design packages and design subsystems with well-defined interfaces, representing what will become components in the implementation. It also contains descriptions of how objects of these design classes collaborate to perform use cases. The next figure shows part of a sample design model for the recycling-machine system in the use-case model shown in the previous figure.



Part of a design model with communicating design classes and package group design classes.

The design activities are centered around the notion of architecture. The production and validation of this architecture is the main focus of early design iterations. Architecture is represented by a number of architectural views [9]. These views capture the major structural design decisions. In essence, architectural views are abstractions or simplifications of the entire design, in which important characteristics are made more visible by leaving details aside. The architecture is an important vehicle not only for developing a good design model, but also for increasing the quality of any model built during system development.

Implementation

The purpose of implementation are:

- To define the organization of the code, in terms of implementation subsystems organized in layers.
- To implement classes and objects in terms of components (source files, binaries, executables, and others).
- To test the developed components as units.
- To integrate the results produced by individual implementers (or teams), into an executable system.

The system is realized through implementation of components. The Rational Unified Process describes how you reuse existing components, or implement new components with well defined responsibility, making the system easier to maintain, and increasing the possibilities to reuse.

Components are structured into Implementation Subsystems. Subsystems take the form of directories, with additional structural or management information. For example, a subsystem can be created as a directory or a folder in a file system, or a subsystem in Rational/Apex for C++ or Ada, or packages using Java.

Test

The purposes of testing are:

- To verify the interaction between objects.
- To verify the proper integration of all components of the software.
- To verify that all requirements have been correctly implemented.

- To identify and ensure defects are addressed prior to the deployment of the software.

The Rational Unified Process proposes an iterative approach, which means that you test throughout the project. This allows you to find defects as early as possible, which radically reduces the cost of fixing the defect. Test are carried out along three quality dimensions reliability, functionality, application performance and system performance. For each of these quality dimensions, the process describes how you go through the test lifecycle of planning, design, implementation, execution and evaluation.

Strategies for when and how to automate test are described. Test automation is especially important using an iterative approach, to allow regression testing at then end of each iteration, as well as for each new version of the product.

Deployment

The purpose of the deployment workflow is to successfully produce product releases, and deliver the software to its end users. It covers a wide range of activities including:

- Producing external releases of the software
- Packaging the software
- Distributing the software
- Installing the software
- Providing help and assistance to users

In many cases, this also includes activities such as:

- Planning and conduct of beta tests
- Migration of existing software or data
- Formal acceptance

Although deployment activities are mostly centered around the transition phase, many of the activities need to be included in earlier phases to prepare for deployment at the end of the construction phase.

The Deployment and Environment workflows of the Rational Unified Process contain less detail than other workflows.

Project Management

Software Project Management is the art of balancing competing objectives, managing risk, and overcoming constraints to deliver, successfully, a product which meets the needs of both customers (the payers of bills) and the users. The fact that so few projects are unarguably successful is comment enough on the difficulty of the task.

This workflow focuses mainly on the specific aspect of an iterative development process. Our goal with this section is to make the task easier by providing:

- A framework for managing software-intensive projects
- Practical guidelines for planning, staffing, executing, and monitoring projects
- A framework for managing risk

It is not a recipe for success, but it presents an approach to managing the project that will markedly improve the odds of delivering successful software. [14]

Configuration and Change Management

In this workflow we describe how to control the numerous artifacts produced by the many people who work on a common project. Control helps avoid costly confusion, and ensures that resultant artifacts are not in conflict due to some of the following kinds of problems:

- Simultaneous Update--When two or more workers work separately on the same artifact, the last one to make changes destroys the work of the former.
- Limited Notification--When a problem is fixed in artifacts shared by several developers, and some of them are not notified of the change.
- Multiple Versions--Most large programs are developed in evolutionary releases. One release could be in customer use, while another is in test, and the third is still in development. If problems are found in any one of the versions, fixes need to be propagated between them. Confusion can arise leading to costly fixes and re-work unless changes are carefully controlled and monitored.

This workflow provides guidelines for managing multiple variants of evolving software systems, tracking which versions are used in given software builds, performing builds of individual programs or entire releases according to user-defined version specifications, and enforcing site-specific development policies.

We describe how you can manage parallel development, development done at multiple sites, and how to automate the build process. This is especially important in an iterative process where you may want to be able to do builds as often as daily, something that would become impossible without powerful automation. We also describe how you can keep an audit trail on why, when and by whom any artifact was changed. This workflow also covers change request management, i.e. how to report defects, manage them through their lifecycle, and how to use defect data to track progress and trends.

Environment

The purpose of the environment workflow is to provide the software development organization with the software development environment--both processes and tools--that are needed to support the development team.

This workflow focuses on the activities to configure the process in the context of a project. It also focus on activities to develop the guidelines needed to support a project. A step-by-step procedure is provided describing how you implement a process in an organization.

The environment workflow also contains a Development Kit providing you with the guidelines, templates and tools necessary to customize the process. The Development Kit is described in more detail in the section " Development Kit for Process Customization" found later in this paper.

Certain aspects of the Environment workflow are not covered in the process such as selecting, acquiring, and making the tools work, and maintaining the development environment.

Rational Unified Process--The Product

The Rational Unified Process product consists of:

- A web-enabled searchable knowledge base providing all team members with guidelines, templates, and tool mentors for all critical development activities. The knowledge base can further be broken down to:
  - Extensive guidelines for all team members, and all portions of the software lifecycle. Guidance is provided for both the high-level thought process, as well as for the more tedious day-to-day activities. The guidance is published in HTML form for easy platform-independent access on your desktop.
  - Tool mentors providing hands-on guidance for tools covering the full lifecycle. The tool mentors are published in HTML form for easy platform-independent access on your desktop. See section "Integration with Tools" for more details.

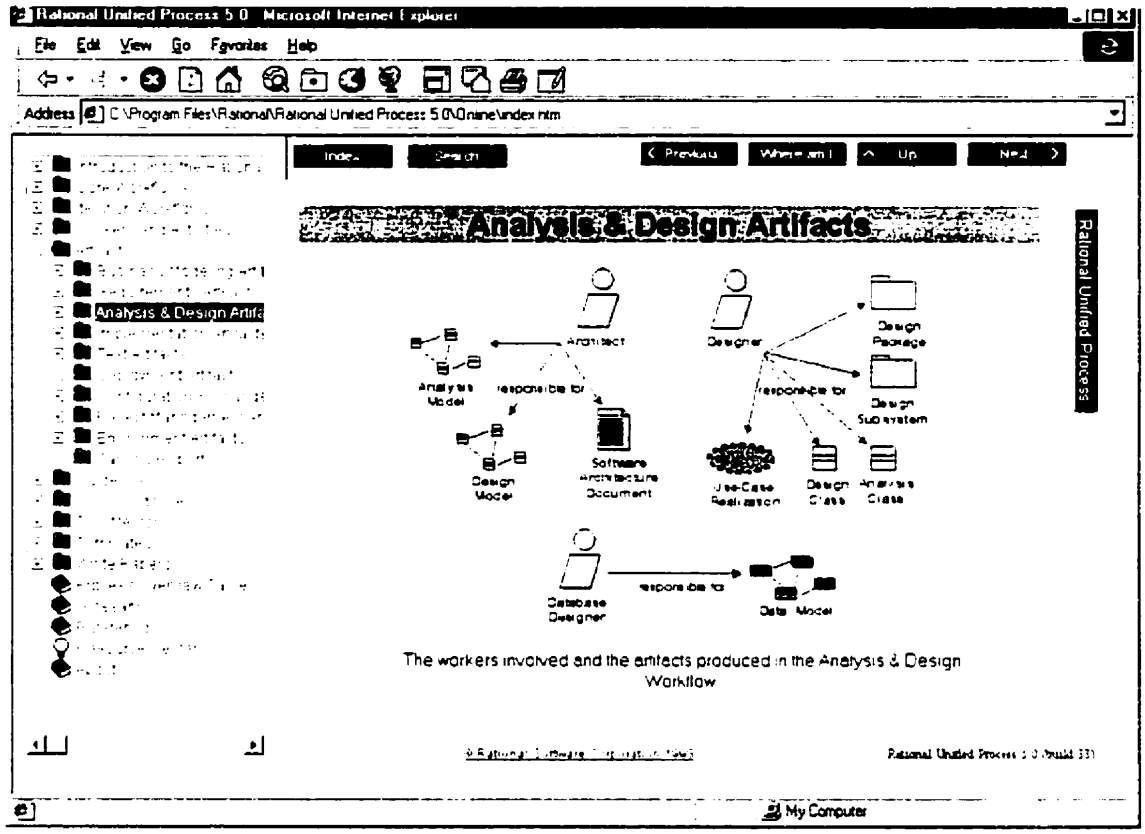
    - Rational Rose® examples and templates providing guidance for how to structure the information in Rational Rose when following the Rational Unified Process (Rational Rose is Rational's tool for visual modeling)
    - SoDA® templates--more than 10 SoDA templates that helps automate software documentation (SoDA is Rational's Document Automation Tool)
    - Microsoft® Word templates--more than 30 Word templates assisting documentation in all workflows and all portions of the lifecycle
    - Microsoft Project Plans--Many managers find it difficult to create project plans that reflects an iterative development approach. Our templates jump start the creation of project plans for iterative development, according to the Rational Unified Process.
    - Development Kit--describes how to customize and extend the Rational Unified Process to the specific needs of the adopting organization or project, as well as provides tools and templates to assist the effort. This development kit is described in more detail later in this section.
    - Access to Resource Center containing the latest white papers, updates, hints, and techniques, as well as references to add-on products and services.
- A book *Rational Unified Process--An Introduction*, by Philippe Kruchten, published by Addison-Wesley. The book is on 277 pages and provides a good introduction and overview to the process and the knowledge base.

Navigating the Knowledge Base

The Rational Unified Process knowledge allows you to access the content with any of the popular web browsers, such as Microsoft® Internet Explorer and Netscape Navigator.

With the Rational Unified Process, you're never more than a few mouse clicks away from the information you want. The knowledge base contains a lot of hypertext links, and overviews of the various process elements are presented through interactive images, making it easy to find relevant information in an intuitive fashion. The powerful search engine, the index, and the 'explorer looking' tree browser make it easy to use the process. Navigational buttons allow you to move to the next or previous page as if reading a book.

Information is presented in many different views, allowing you to look at information relevant to your role, to a specific activity, or to a workflow. Guided tours for easy learning of the process are provided for key project roles.



Interactive images and navigational buttons make it easy to find the specific information you are looking for.

Development Kit for Process Customization

The Rational Unified Process is general and complete enough to be used "as is" by some software development organizations. However in many circumstances, this software engineering process will need to be modified, adjusted, and tailored to accommodate the specific characteristics, constraints, and history of the adopting organization. In particular a process should not be followed blindly, generating useless work, producing artifacts that are of little added value. It must be made as lean as possible and still be able to fulfill its mission to produce rapidly and predictably high quality software.

The process contains a Development Kit, which contains guidelines for how you can customize the process to fit the specific needs of the adopting organization or project. Templates are also included for process authoring, as well as tools for generation or manipulation of search engine, index, site map, tree browser, etc. The Development Kit enables the customizing organization to maintain the look and feel of the Rational Unified Process.

The more the process is customized, the more difficult will it be to move over customizations to future releases of the process. The Development Kit describes strategies, tools and techniques to minimize the work associated with moving customizations to future releases.

Integration with Tools

A software-engineering process requires tools to support all activities in a system's lifecycle, especially to support the development, maintenance and bookkeeping of various artifacts-models in particular. An iterative development process puts special requirements on the tool set you use, such as better integration among tools and round-trip engineering between models and code. You also need tools to keep track of changes, to support requirements traceability, to automate documentation, as well as tools to automate tests

to facilitate regression test. The Rational Unified Process can be used with a variety of tools, either from Rational or other vendors. However, Rational provides many well-integrated tools that efficiently support the Rational Unified Process.

Below you find a list of some of Rational's tools that support the Rational Unified Process.
The Rational Unified Process contains Tool Mentors for almost all of these products. A *tool mentor* is a step-by-step guide describing in detail how to operate a tool, (i.e. what menus to launch, what information to enter into dialog boxes, and how to navigate a tool) to carry out an activity within the process. The Tool Mentors allow us to link the tool-independent process to the actual manipulation of the tools in your daily work.

- **Rational Requisite®Pro**--Keeps the entire development team updated and on track throughout the application development process by making requirements easy to write, communicate and change. **Rational ClearQuest™**--A Windows and Web-based change-request management product that enables project teams to track and manage all change activities that occur throughout the development lifecycle. **Rational Rose 98**--The world's leading visual modeling tool for business process modeling, requirements analysis, and component architecture design.
- **Rational SoDA**--Automates the production of documentation for the entire software development process, dramatically reducing documentation time and costs.
- **Rational Purify®**--A run-time error checking tool for application and component software developers programming in C/C++; helps detect memory errors.
- **Rational Visual Quantify™**--An advanced performance profiling tool for application and component software developers programming in C++, Visual Basic, and Java; helps eliminate performance bottlenecks.
- **Rational Visual PureCoverage™** --Automatically pinpoints areas of code not exercised in testing so developers can thoroughly, efficiently and effectively test their applications.
- **Rational TeamTest**--Creates, maintains and executes automated functional tests, allowing you to thoroughly test your code and determine if your software meets requirements and performs as expected.
- **Rational PerformanceStudio™**--An easy-to-use, accurate and scalable tool that measures and predicts the performance of client/server and Web systems.
- **Rational ClearCase®**--Market-leading software configuration management tool, giving project managers the power to track the evolution of every software development project.

A Brief History of the Rational Unified Process

The Rational Unified Process has matured over many years and reflects the collective experience of the many people and companies that make up today Rational Software's rich heritage.

Let us have a quick look at the process's ancestry, as illustrated in the figure below.

Genealogy of the Rational Unified Process

Going backwards in time, the Rational Unified Process is the direct successor to the Rational Objectory Process (version 4). The Rational Unified Process incorporates more material in the areas of data engineering, business modeling, project management, and configuration management, the latter as a result of the merger with Pure-Atria. It also brings a tighter integration to the Rational Software suite of tools. The Rational Objectory Process was the result of the integration of the "Rational Approach" and the Objectory Process (version 3), after the merger of Rational Software Corporation and Objectory AB in 1995. From its Objectory ancestry, the process has inherited its process structure and the central concept of use case. From its Rational background, it gained the current formulation of iterative development and architecture. This version also incorporated material on requirements management from Requisite, Inc. and a detailed test process inherited from SQA®, Inc., companies which also merged with Rational Software. Finally, this process was the first one to use the newly created Unified Modeling Language (UML 0.8).

The Objectory process was created in Sweden in 1987 by Ivar Jacobson as the result of his experience with Ericsson. This process became a product at his company, Objectory AB. Centered around the concept of use case and an object-oriented design method, it rapidly gained recognition in the software industry and has been adopted and integrated by many companies worldwide. A simplified version of the Objectory process was published as a text book in 1992.

The Rational Unified Process is a specific and detailed instance of a more generic process described by Ivar Jacobson, Grady Booch, and James Rumbaugh in the textbook, *The Unified Software Development Process*.

References

1. Barry W. Boehm, *A Spiral Model of Software Development and Enhancement*, Computer, May 1988, IEEE, pp.61-72
2. Barry W. Boehm, *Anchoring the Software Process*, IEEE Software, 13, 4, July 1996, pp. 73-82.
3. Grady Booch, Object Solutions, Addison-Wesley, 1995.
4. Grady Booch, Ivar Jacobson, and James Rumbaugh, *Unified Modeling Language 1.3*, White paper, Rational Software Corp., 1998.
5. Alan W. Brown (ed.), *Component-Based Software Engineering*, IEEE Computer Society, Los Alamitos, CA, 1996, pp.140.
6. Michael T. Devlin, and Walker E. Royce, Improving Software Economics in the Aerospace and Defense Industry, Technical paper TP-46, Santa Clara, CA, Rational Software Corp., 1995
7. Ivar Jacobson, Magnus Christerson, Patrik Jonsson, and Gunnar Övergaard, *Object-Oriented Software Engineering-A Use Case Driven Approach*, Wokingham, England, Addison-Wesley, 1992, 582p.
8. Ivar Jacobson, M. Griss, and P. Jonsson, *Software Reuse-Architecture, Process and Organization for Business Success*, Harlow, England, AWL, 1997.
9. Philippe Kruchten, *The 4+1 View Model of Architecture*, IEEE Software, 12 (6), November 1995, IEEE, pp.42-50.
10. Philippe Kruchten, *A Rational Development Process*, CrossTalk, 9 (7), STSC, Hill AFB, UT, pp.11-16.
11. Ivar Jacobson, Grady Booch, and Jim Rumbaugh, *Unified Software Development Process*, Addison-Wesley, 1999.
12. Grady Booch, Jim Rumbaugh, and Ivar Jacobson, *Unified Modeling Language-User's Guide*, Addison-Wesley, 1999.
13. Philippe Kruchten, *Rational Unified Process-An Introduction*, Addison-Wesley, 1999.
14. Walker Royce, *Software Project Management-A Unified Framework*, Addison-Wesley, 1998.

ANNEXE 7

Présentation - Révision des processus

R&D
Process Software Review

September 9, 1999

TECSYS

Agenda

- Introduction
- Process Software Architect Role
- Software development process
- Overview of Rational Unified Process (RUP)
- Pilot Project
- Rational Tools
- Resource Involvement
- Timetable for the implementation/training
- Miscellaneous

Introduction

- Last Year in R&D
  - BPD
  - New Organizational Chart
  - Implementation of new processes

Agenda

- Introduction
- Process Software Architect Role
- Software development process
- Overview of Rational Unified Process (RUP)
- Pilot Project
- Rational Tools
- Resource Involvement
- Timetable for the implementation/training
- Miscellaneous

Process Software Architect Responsibilities

- Review and evaluate current process
- Implement new processes using RUP framework
- Ensure a standardization in
  - Tools
  - Documentation of process/activities
- Assist resources when evaluating new tools
- Training on new processes/tools
- Change management / Communication Plan

What is change management?

- Activities related to communication, training and the impacts of change.
- The impacts are defined as the gap between the current state and the desired state.
- Change management aims at identifying those impacts, to measure them and find means to help resources, processes and methods to reach the desired state.

Change Management

Identify Stakeholders

- Who might influence what we are doing?
- Who will be impacted by what we are doing?

- Identify the owners of the processes
- Identify resources impacted by the changes
  - Put in place a change management plan
  - Deal with change resistance
- Put in place a communication plan

Agenda

- Introduction
- Process Software Architect Role
- Software development process
- Overview of Rational Unified Process (RUP)
- Pilot Project
- Rational Tools
- Resource Involvement
- Timetable for the implementation/training
- Miscellaneous

Software development process

- Help to direct the tasks of individual developers and the team as a whole
- Guides the efforts of the people involved in a software project by providing template that explains the steps needed to complete the project

Software development process

- Specify which and when artifacts should be developed
- Offer criteria for monitoring and measuring the project's products and activities

Software development process

- Impact of not having a well-defined process:
  - Unhappy customers
  - Unhappy resources
  - $$$$$$ Expensive

Who are R&D Customers

Customer	Needs
Sales&Marketing	Clear vision of the releases
C/E	A well designed software that can be easily modified.
Professional Services	Good features. Stable product with no bugs.
Installation	A product that can be installed. Well tested technologically. Informix Version, tools etc.

Impact on Resources

Without a well-defined process the team will develop in an ad hoc manner, with success relying on the heroic efforts of a few dedicated individual.

Agenda

- Introduction
- Process Software Architect Role
- Software development process
- Overview of Rational Unified Process (RUP)
- Pilot Project
- Rational Tools
- Resource Involvement
- Timetable for the implementation/training
- Miscellaneous

Overview of RUP

- What is RUP (Rational Unified Process)

  The Rational Unified Process is a software engineering process.

  Provides a generic process framework that can be specialized for a very large class of software systems, for different application areas and different project sizes.

RUP Key Concepts

- The **process** is describe in term of workflow.
- A **workflow** is a sequence of activities, which are ordered so that one produces an output that works as an input for the next activity.
- An **activity** is something that a worker does that provides a meaningful result in the context of a project. The activity must have a clear purpose.
- An **artifact** is a work product of the process. Artifacts are the responsibility of a single worker.

RUP Key Concepts

- We need to promote the idea that every piece of information in the process must be the responsibility of a specific person.
- A **worker** defines the behavior and responsibilities of an individual, or a set of individuals working together as a team. The worker represents a role played by an individual on a project, and defines how they carry out work.
- We need to identify the skills requirement for each kind of worker in our process.

Workflow

- Major workflow provided in RUP
  - Process workflows
    - Business Modeling
    - Requirements workflow
    - Analysis and Design workflow
    - Implementation workflow
    - Test workflow
    - Deployment workflow
  - Supporting workflows
    - Configuration Management
    - Project Management
    - Environment

Requirements workflow

- Defining functionality of the system
- Planning the technical contents of iterations
- Provide a basis for estimating cost and time to develop the system (release)
- Define a user interface for the system

Analysis/design workflow

- Translate the requirements (functional analysis) into specification
- Selecting the best implementation strategy
- Database design
- To evolve a robust architecture for the system

Implementation workflow

- Implementation in terms for components: source code, scripts, executable, and the like.
- Unit-test the components
- Integrate the results produced by individual implementers (or teams), into an executable system.

Test workflow

- To verify the proper integration of all components of the software.
- To verify that all requirements have been correctly implemented.
- To identify and ensure defects are addressed prior to the deployment of the software.

Deployment workflow

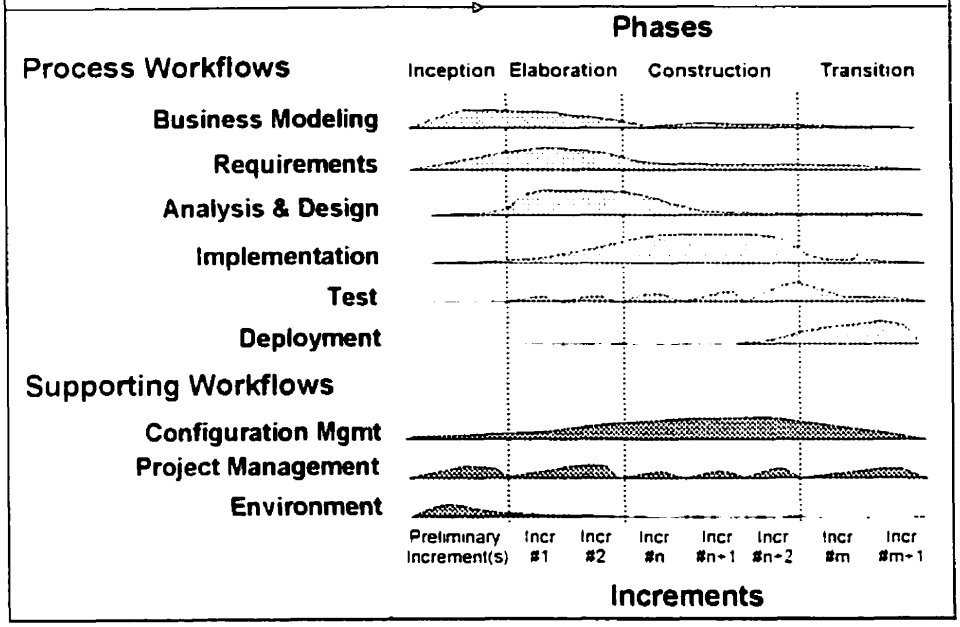
The purpose of the deployment workflow is to successfully produce product releases, and deliver the software to its end users.

- Packaging the software
- Distributing the software
- Installing the software
- Providing help and assistance to users
- Training the users

RUP Phases

- **Inception**
  - Establish business case & project scope for the project
- **Elaboration**
  - Analyze the problem, establish sound architectural foundation, eliminate highest risk elements of project, establish project plan
- **Construction**
  - Building the product as a series of incremental iterations
  - Emphasis on resources, quality, costs, schedules
- **Transition**
  - Transition the software into user community (beta testing, training, rollout, etc)

Unified Process



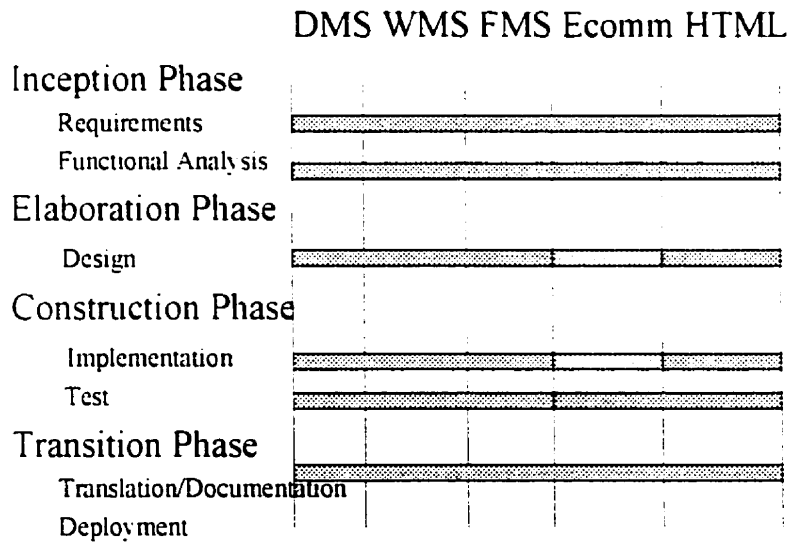
Agenda

- Introduction
- Process Software Architect Role
- Software development process
- Overview of Rational Unified Process (RUP)
- Pilot Project
- Rational Tools
- Resource Involvement
- Timetable for the implementation/training
- Miscellaneous

Implementing RUP at Tecsys

- Tecsys environment:
  - multiple products. DMS-WMS-FMS-EliteView- EliteQ-E.Comm
  - multiple technologies: Informix-Java-HTLM-Oracle
- Customize RUP per development environment / Maximize common ground
- Incremental approach to implementation

Development Environment



Where do we start?

Start from the beginning

- Inception Phase across R&D for 6.4
  - Requirements Management
  - Functional Analysis using use cases for one or two projects only. The full use of use cases for 6.5.
- Workflows: Requirement - Part of Design
- Evaluation of Requisite-Pro

Pilot Project

- Why?
  - Fast benefit: Better planning and scoping of a release.
  - Planning Features and Technology earlier in the process.
  - Assess risk earlier in the process.
  - Positive Impact on other workflow
    - Introduction of a requirement management tool
    - Use case -> benefit for design and test plan

Agenda

- Introduction
- Process Software Architect Role
- Software development process
- Overview of Rational Unified Process (RUP)
- Pilot Project
- Rational Tools
- Resource Involvement
- Timetable for the implementation/training
- Miscellaneous

Rational Tools

- Requisite-Pro: Requirements management tool.
  - Capture/prioritize/follow-up requirements
  - Follow requirements during the project life cycle. (Requirements-Functional analysis - Design - Test Plan) **Traceability.**

    The purpose of establishing **traceability** is to help
    - Verify that all requirements of the system are fulfilled by the implementation.
    - Verify that the application does only what it was intended to do.
    - Manage change.

Rational Tools

Requisite-Pro

- Traceability helps you follow how system features as specified in the vision document. These in turn are detailed into use cases and other requirements. You then need to follow how these detailed specifications are translated into a design, how it is tested, and how it is documented for the user.

Rational Tools

Rational Test Suite

Rational Robot - for thorough functional testing of your entire application.

Rational TestFactory--automatically detects runtime errors without user assistance and generates optimal scripts for regression testing.

Rational Purify--locates hard-to-find runtime errors that cause program crashes.

Rational Visual Quantity--pinpoints performance bottlenecks in your applications.

Rational Visual PureCoverage -- identifies untested code and provides code-coverage analysis.

Rational Tools

Rational Unified Process -- a comprehensive knowledge base of software development best practices.

Rational ClearQuest--e-mail enabled and Web-accessible defect tracking and change request management.

Rational SoDA--automated reporting and documentation.

Agenda

- Introduction
- Process Software Architect Role
- Software development process
- Overview of Rational Unified Process (RUP)
- Pilot Project
- Rational Tools
- Resource Involvement
- Timetable for the implementation/training
- Miscellaneous

Resource Involvement

- Need to set up a working group for the first implementation of the Inception Phase:
  - Product manager
  - Analyst
  - Architect
  - Test
  - Documentation

Agenda

- Introduction
- Process Software Architect Role
- Software development process
- Overview of Rational Unified Process (RUP)
- Pilot Project
- Rational Tools
- Resource Involvement
- Timetable for the implementation/training
- Miscellaneous

Timetable for Phase 1

- September:
  - Demo and evaluation of Requisite-Pro
- October:
  - Training on Requisite-Pro
  - Training on the RUP Inception Phase
- November - December
  - Customization of the RUP Requirement workflow
  - Revising templates

After Phase 1

- Continue implementing RUP by phases
- On going process improvement

Agenda

- Introduction
- Process Software Architect Role
- Software development process
- Overview of Rational Unified Process (RUP)
- Pilot Project
- Rational Tools
- Resource Involvement
- Timetable for the implementation/training
- Miscellaneous

Miscellaneous

- Other Project in Software Process
  - Implementation of a new time job entry
    - For November
  - Evaluating the Rational Testing Tool
  - Evaluating the Rational Documentation Tool
  - Evaluating the Rational Bug tracking system
  - Standardization for documenting our process and activities

Key factor to success

- Support from Top Management and it must trickle down
- Resource involvement
- Communication
- Stay realistic

ANNEXE 8

Gestion des exigences

TECSYS

Research & Development

Process Improvement

Requirements Workflow

Revision 1.0 - 1

Sept 3, 1999

Table of Content

Revision History - Approvers

Revision	Date	Author	Revision Summary	Approver	Approval Date	Note
1.0	Sept 3, 1999	Lucie St-Germain				

1. Introduction

The objective of this document is to determine the differences between the requirement workflow suggested by RUP and the current R&D process in this area.

1.1 Scope

This document highlights the differences between RUP and the current R&D process for requirement management and suggests an action plan to adapt the R&D process.

1.2 Definitions, Acronyms and Abbreviations

RUP	Rational Unified Process

1.3 References

N/A

2. Requirement Workflow

2.1 Summary

The purposes of requirements as per RUP are:

- To come to an agreement with the customer and the users on what the system should do.
- To give system developers a better understanding of the requirements on the system.
- To delimit the system.
- To provide a basis for planning the technical contents of iterations.
- To define a user-interface for the system.

To achieve these goals:

- Stakeholder requests are gathered and analyzed to get a "wish list" of what different stakeholders of the project expect the system to include, together with information on how each request has been considered by the project.
- A vision document is written to describe the overall goals of the project and main features of the system.
- A use-case model is developed
- Use cases and supplementary specification are developed that describes what the system will do

Complementary to the above mentioned artifacts, the following artifacts are developed:

- Glossary
- Use-Case Storyboard
- User-Interface Prototype

2.2 Workflow activities

Here is a list of activities included in the Requirement workflow. The activities are detailed by type of worker.

System Analyst: Lads and coordinates requirements elicitation and use-case modeling, by outlining the system's functionality and delimiting the system. The system analyst is responsible to:

- Develop Vision
- Elicit Stakeholder Requests
- Manage Dependencies
- Capture a common vocabulary
- Find Actors and Use Cases

- Structure the Use-Case Model

Architect: The Architect leads and coordinates technical activities and artifacts throughout the project. The Architect establishes the overall structure

- Prioritize Use-Cases

Use-Case specifier: The use-case specifier details the specification of a part of the system's functionality by describing the Requirements aspect of one or several use cases and other supporting software requirements.

- Detail a use-case
- Detail the software requirements

User Interface Designer: The user-interface designer leads and coordinates the prototyping and design of the user interface

- Model the user-interface
- Prototype de user-interface

Requirements reviewer: The requirements reviewer plans and conducts the formal review of the use-case model.

- Review requirements

2.3 Workflow artifacts

Glossary: The Glossary defines important terms used in the project.

Vision: The Vision is a general vision of the core project's requirements, and provides the contractual basis for the more detailed technical requirements.

Stakeholder request: This artifact contains any type of requests a stakeholder (customer, end user, marketing person, and so on) might have on the system to be developed. It also contains references to any type of external sources to which the system must comply.

Supplementary specifications: The Supplementary Specifications capture the system requirements that are not readily capturable in the use cases of the use-case model.

Change request: Changes to development artifacts are proposed through Change Requests (CRs). Change Requests are used to document and track defects, enhancement requests and any other type of request for a change to the product. The benefit of Change Requests is that they provide a record of decisions, and, due to their assessment process, ensure that change impacts are understood project wide.

2.4 Workflow detail

To help explain the Requirements workflow, we have organized the activites and artifacts in six workflow details:

1. Analyze the problem: The purpose is to produce the vision document for the project and to agree on features and goals of the system.

2. Understand stakeholder needs: The purpose is to collect and elicit information from stakeholders the project. The collected stakeholder requests as a "wish list" that will be used as primary input to defining the use-case model, use cases and supplementary specifications.

3. Define the system: The purpose is to:

   - Align the project team in their understanding of the system.
   - Perform a high-level analysis on the results of collecting stakeholder requests.
   - More formally document the results in models and documents.

4. Manage the scope of the project: The purpose of this workflow detail is to:

   - Define input to the selection of requirements that are to be included in the current iteration.
   - Define the set of features and use cases (or scenarios) that represent some significant, central functionality.
   - Define which requirement attributes and traceabilities to maintain.

5. Refine the system definition: The purpose is to further refine the requirements by describing the use case's flow of event in detail, by modeling the user interface and by detailing the supplementary specifications.

6. Manage changing requirements: The purpose is to:

   - Evaluate formally submitted change requests and determine their impact on the existing requirement set.
   - Structure the use-case model.
   - Set up appropriate requirements attributes and traceabilities.
   - Formally verify that the results of the Requirements workflow conform to the customer's view of the system.

3. Comparison with current process

Lets analyze each sub-workflow and compare to the current R&D activities.

1. Analyze the problem: The purpose is to produce the vision document for the project and to agree on features and goals of the system.

Currently in R&D a business requirements document is produce by the product direction team. This document does not have a standard format and doesn't have all the information necessary to take the right decision for the development solution in Elite. These documents, when produced, often give the solution but not clearly state the problem that needs to be solved. At that level the problem needs to be detailed, it is yet too early to talk about the solution.

2. Understand stakeholder needs: The purpose is to collect and elicit information from stakeholders the project. The collected stakeholder requests as a "wish list" that will be used as primary input to defining the use-case model, use cases and supplementary specifications.

The stakeholders at Tecsys are the marketing division, services division, pre-sales and sales and our end-user. Product direction is constantly in communication with marketing and the sales force. Product direction attends the executive customer meeting that is being held once per year. Product direction attends multiple seminars where Tecsys is present in order to gather the feel for the market and for the customer satisfaction. These are all ways to collect and elicit information from stakeholders. This activity is quite under control. A product committee is in place to decide through all information gathered the direction of the product. The plan decided by the product committee will derive the content for the future releases of the product and triggers the production of business requirements. A functional analysis document is produced based on the business requirements. But currently in R&D we don't use use-case model and use cases for our 4G/L development to detail the functionality. Only our Java development uses these techniques.

3. Define the system

A functional analysis document is produce.

4. Manage the scope of the project

The solution for a particular business requirement is discussed with product direction and a decision is taken on the scope of the project. The scope of an entire release is not manage, only individual projects.

5. Refine the system definition

This activity is not being done in R&D. We define the user-interface muck-up at the design level only and not at the functional level.

6. Manage changing requirements

The requirements changed throughout the project without any impact analysis being done. The changes are not coordinated between different groups and are not documented.

4. IMPLEMENTATION IN R&D

The elicitation of needs from our stakeholders is very much in control in R&D and there is not much improvement we could implement at that level. The communication of these needs between product direction and the development team must be improved. Studying the vision document suggested from RUP and looking at our business requirement document produced in R&D. our first suggestion is to customize and implement the vision document.

Afterwards we need to replace our functional analysis document in order to use a customize version of the Software Requirement Specifications from RUP. This SRS will be produce in multiple iteration. The first iteration of this document will be to find all use cases for a given solution (independent of our development environment 4G/L or Java) and estimate the effort of this solution in order to scope earlier the project. At this time we will introduce the concept of an architect to prioritize the use-cases and to participate in the effort estimate. If necessary the analyst will be asked to detail some use-cases that are identified had being a greater risk.

Once the solution and the effort estimate has been approved then the second iteration of the SRS will be to detail the major use-cases and to product screen muck-up. Once the detailed use-cases and the user-interface are accepted then a designer will produce the design of the solution. (Changes to function, database, classes)

ANNEXE 9

Document de vision

TECSYS

Research & Development

This is our typical Vision document template. If you need for your project any one of the following sections then you must use the full Vision document template

Product positioning statement section

Detailed information on stakeholders and users

Section on competitors

Section on special documentation required

Note: Text enclosed in square brackets and displayed in blue italics (style: InfoBlue) is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal style: Body Text.]

Note: If a section does not apply then you must indicate Non Applicable. Do not delete the section

Vision Document

Project Name

Revision 0.1 - 1

Revision Date (MMM DD, YYYY)

Table of Content

Revision History - Approvers

Revision	Date	Author	Revision Summary	Approver	Approval Date	Note
0.1	Dec 13, 1999	Lucie St-Germain	Initial Draft	Approver 1 Approver 2	Jan 01, 2000	*Done by e-mail, by phone or in person*

Document revision numbers are employed to track the revisions of a document when being used by multiple users parties to ensure all users parties employ the most current revision of the document.

The document revision numbers conform to the following standards:

Revision No: X.XX

"X" is incremented by "1" each time an official release of the document is issued by the team. A release is considered official once the appropriate approval signatures, eg.: Manager or Manager and Director have approved the document for distribution either to external teams or within their teams.

***"XX"** is incremented by "1" each time an internal (within the team) or working copy of the document is circulating within a team pending an official release by the team issuing party. This allows the team to track minor modifications to the document, internally within the team, pending a new or another official release by the issuing team parties.*

Date format should be MMM DD, YYYY (example Jul 8, 2000)

Use the full name under the Author and Approver headings – no initials.

Under Revision Summary, explain in simple general terms what you changed in the document since the previous revision.

Under Approver column, list all people that must approve the Vision. One line per approver.

The signature is not needed, under note explain if the approval was done by e-mail, by phone or in person.

Note: When changing the revision, turn on the History in Word (Menu: Tool/Track Changes/Highlight Changes) to be able to visualize the changes since the last version.

1. Introduction (Required)

The objective of this document is to collect, analyze and define high-level needs of the <<*Feature name*>> feature. It focuses on the capabilities needed by the stakeholders, and the target users, and why these needs exist. The details of how the <<*Feature name*>> feature fulfils these needs are detailed in the functional analysis document.

1.1 Purpose (Required)

[Specify the purpose of this Vision. This section must be filled and must explain the objective of this change

Ex: The purpose for this document is to enable ABC capability for the XYZ product(s).

Ex: Support Oracle Database with our DMS product

Ex: To provide the ability to the distributors to add a cost when transfers occur between two warehouses

Ex: Convert our front-end to HTML for the EMS product.

Ex: To provide the ability to the distributor to maintain the discounts negotiated with their vendors in ***Elite****Series P.O application. That is, to improve our P.O functionality.]*

1.2 Scope (Required)

[A brief description of the scope of this Vision; what Project(s) it is associated with, and anything else that is affected or influenced by this document.]

1.3 Definitions, Acronyms and Abbreviations (Required)

[This subsection should provide the definitions of all terms, acronyms, and abbreviations required to interpret the Vision properly. This information may be provided by reference to the project Glossary.]

When we introduce a new concept, this section will contain the definitions of the new terminology. When abbreviations are used in this document then they also must be written in this section. (New or old ones) Must be sorted in alphabetical order.

Ex

API	*Application Program Interface*
Bill-Back discount	*Discount that is collected from the vendor at a later time. Also referred to as Rebate*
OMS	*Order Management System*
SPO	*Special Order*

Note: When a new acronym is introduced you must use the full name with the acronym in parentheses for the first occurrence. Afterwards, use only the acronym

1.4 References (Optional)

[References to any other document that may help the understanding of the Vision document must be written in this section. If this document is part of a series of documents then this should be mentioned in this section.

Ex: Research document. A custom quote done by C.E. A Web site]

2. Positioning

2.1 Business Opportunity (Optional)

[Briefly describe the business opportunity being met by this project]

2.2 Problem Statement (Required)

[Provide a statement summarizing the problem being solved by this project. Explain the business problem or scenario that needs to be solved with this project. Express without referring to **Elite**Series*. The following format may be used.]*

The problem of	(describe the problem)
Affects	(the stakeholders affected by the problem).
The impact of which is	(what is the impact of the problem).
A successful solution would	(list some key benefits of a successful solution).

3. Stakeholder and User Descriptions (Optional)

3.1 Stakeholder Summary

[Present a summary list of all the identified stakeholders:]

Name	Represents	Role
Name the stakeholder type	Briefly describe what they represent with respect to the development. Ex. Sales, Marketing, Pre-sales, Product Direction	Briefly describe the role they are playing in the development i.e. Ensure this...

3.2 User Summary

Present a summary list of all the identified users:

Name	Description
Name the user type *Ex. Buyer, Picker, Order Entry clerk*	Briefly describe what they represent with respect to the system.

4. Product Overview (Required)

[This section consists of two subsections, as follow:

Product perspective puts the project in perspective with other related products

Assumptions and dependencies]

4.1 Product Perspective (Required)

[This subsection of the Vision document should put the product project in perspective with other related products and the user's environment. If the product is independent and totally self-contained, state it here. If the product is a component of a larger system, then this subsection should relate how these systems interact and should identify the relevant interfaces between the systems.]

*This subsection identifies the products affected by the feature being added in Elite*Series

Product	Affected	Comment
Elite*Series* DMS		
Elite*Series* WMS		
Elite*Series* FMS		
Control*Series FMS*		
Control*Series PFMS*		
EDI		
Elite eCom		
Buyer's Tool		
EliteQ		
Elite*View*		
You may enter another product not specified		

Note: In the event that further investigation by the analyst reveals an unanticipated impact on other products, it is the responsibility of the analyst to update this section to reflect the change in impact.

4.2 Assumptions and Dependencies (Optional)

[List each of the factors that affect the features stated in the Vision document. List assumptions that, if changed, will alter the Vision document. For example, an assumption may state that a specific operating system will be available for the hardware designated for the software product. If the operating system is not available, the Vision document will need to change.]

Example: if a project depends on having another project completed.

Ex: Special orders must consider the changes done by the catalog modifications.

4.3 Cost (Required)

Labor budget fixed by Product Direction: $$$

Others: (ex: equipment, special training, software, etc.)

4.4 Licensing and Installation (Optional)

[Licensing and installation issues can also directly impact the development effort. For example, the need to support serializing, password security or network licensing will create additional requirements of the system that must be considered in the development effort.

Installation requirements may also affect coding, or create the need for separate installation software.]

This section could be used if the vision implicates the use of a third-party software and or special installation that must be taken into account. Ex: HTML project that may need special installation procedures etc. Another project that may need the licensing of a third-party software.

4.5 Upgrade Path (Optional)

Do we need to provide an upgrade path for the existing solution software offering?

It may not be applicable (ex: new product)

For new features added in EliteSeries, an upgrade path is required and it must be stated here. However if you foresee a problem then you must explain it.

5. Product Features (Required)

[List and briefly describe the product features. Features are the high-level capabilities of the system that are necessary to deliver benefits to the users. Each feature is an externally desired service that typically requires a series of inputs to achieve the desired result. For example, a feature of a problem tracking system might be the ability to provide trending reports. As the use-case model takes shape, update the description to refer to the use cases

Because the Vision document is reviewed by a wide variety of involved personnel, the level of detail should be general enough for everyone to understand. However, enough detail should be available to provide the team with the information they need to create a use-case model

Throughout this section, each feature should be externally perceivable by users, operators or other external systems. These features should include a description of functionality and any relevant usability issues that must be addressed

The following guidelines apply:

Avoid design. Keep feature descriptions at a general level. Focus on capabilities needed and why, (not how) they should be implemented.

No reference to tables. No reference to options name.

Use the format: "Ability to".

Prioritize the requirements. (Ex: Necessary and nice to have)]

5.1 Features grouping 1

5.2 Features grouping 2

6. Business scenario (Required)

Briefly describe the business scenario in which this feature will be used at customers sites

Detail and or illustrate the flow description. Do not describe the flow in ***Elite***Series *but the business process flow*

7. Stipulation (Optional)

[Note any design and or applicable technical or business constraints.]

8. Other Product Requirements (Required)

- Application Standards (Optional)

[List all standards the product must comply with. These can include legal and regulatory (FDA, UCC) communications standards (TCP IP, ISDN), platform compliance standards (Windows, Unix, etc), quality and safety standards (UL, ISO, CMM).]

Must follow Tecsys's Development Standards Guide.

EX: EDI standards

- System Requirements (Optional)

[Define any system requirements necessary to support the application. These can include the supported host operating systems and network platforms, configurations, memory, peripherals and companion software.]

- Performance Requirements (Required)

[Use this section to detail performance requirements. Performance issues can include such items as user load factors, bandwidth or communication capacity, throughput, accuracy, reliability or response times under a variety of loading conditions.]

- Environmental Requirements (Optional)

[Detail environmental requirements as needed. Environmental factors can include usage conditions, user environment, resource availability, maintenance issues, error handling and recovery.]

9. Target Release (Optional)

[Records the intended product version in which the feature will first appear.]