SYSTÈME D'ÉVALUATION ET DE GESTION DES RISQUES D'INONDATION EN MILIEU FLUVIAL

PROJET SEGRI

Rapport d'étape #2

Rapport de recherche No R-720-e2 *Janvier 2005*

Système d'**É**valuation et de **G**estion des **R**isques d'**I**nondation en milieu fluvial

Projet SEGRI

Rapport d'étape #2

Présenté au

Fonds des Priorités Gouvernementales en Science et en Technologie – volet Environnement (FPGST-E)

18 janvier 2005

Équipe de réalisation

Institut National de la Recherche Scientifique – Eau, Terre et Environnement

Yves Secretan	Professeur, PhD
Eric Larouche	Ingénieur en informatique

Collaborateurs

Laurent Bonnifait	Étudiant à la maîtrise
Khalid Rahman	Étudiant à la maîtrise
Maude Giasson	Stagiaire, étudiante à la maîtrise
Kevin Solinski	Stagiaire
Cédric Caron	Stagiaire
Benjamin Behaghel	Stagiaire
Daniel Nadeau	Stagiaire
Stéphane Lévesque	Stagiaire
Vincent Martineau	Stagiaire
Sébastien Labbé	Stagiaire
Dave Guérin	Stagiaire

Pour les fins de citation : **Secretan Y., Larouche E. & coll. (2004)**.

Système d'Évaluation et de Gestion des Risques d'Inondation en milieu fluvial (SEGRI) : Rapport d'étape #2. Québec, INRS-Eau, Terre & Environnement. 45 pages. (INRS-Eau, Terre & Environnement, rapport de recherche 720 e2)

Pour: Fonds des Priorités Gouvernementales en Science et en Technologie – volet Environnement (FPGST-E).

ISBN : 2-89146-526-1

Tables des matières

1 Introduction

Ce second rapport couvre la période allant du 5 janvier au 23 décembre 2004 et conclut la seconde année du projet SEGRI. L'objectif de ce rapport d'étape est de faire état de l'avancement du projet.

Une pièce tangible de tous les efforts de développement accomplis jusqu'ici est sans aucun doute Modeleur2 version alpha, le premier livrable logiciel du projet SEGRI. Loin d'être finale, voire complètement fonctionnelle, cette version tente de montrer au maximum les nouvelles fonctionnalités présentes dans le logiciel (séries, infrastructure des partitions, zone d'étude, plug-in, etc.), parfois au détriment de l'aspect esthétique qui sera repris et amélioré en 2005. La plupart des éléments clé du logiciel sont toutefois présents dans cette version. Le document Guide d'installation et description des fonctionnalités de Modeleur2 introduit plus en détails le logiciel, ses fonctionnalités et ses limites actuelles.

Entre autres, l'année 2004 a permis de confirmer les grandes orientations du logiciel qui avait été définies l'année précédente. L'architecture élaborée en 2003 a donc été mise en place avec succès, et tel que prévu elle s'avère être souple et fonctionnelle. Le langage Python qui chapeaute plusieurs sections du logiciel est un choix qui s'est avéré judicieux, car il a permis de mettre en place aisément plusieurs aspects du logiciel (plug-in, interface usager, scripts). Le fait de développer autour d'une base de donnée relationnelle apporte beaucoup en ce qui a trait aux fonctionnalités pour les requêtes spatiales, en plus de valider intrinsèquement la cohérence entre les données. L'utilisation de librairies externes, telles GEOS et VTK, permet la réutilisation de code validé et d'ainsi concentrer les efforts de développement sur ce qui est propriétaire à Modeleur2, plutôt que de réinventer des éléments qui sont disponibles sur le marché.

L'approche consistant à compter sur une équipe de développement constituée de stagiaires a également fait ses preuves. La structure en place au sein du groupe facilite leur intégration et permet de les amener à rapidement contribuer de façon tangible à l'avancement du projet.

Le présent document est constitué de six sections, qui chacune tente de faire ressortir un aspect particulier du développement. La section 1 présente d'abord le contexte dans lequel est réalisé ce développement. La section 3 introduit les spécifications fonctionnelles et fait référence au document Spécifications – Modeleur2. La section 4 décrit en détails quelles ont été les tâches accomplies pendant chaque mois du projet et introduit le document Rapports de recherche 2004 qui est en fait une collection des rapports rédigés en cours d'année. La section 5 présente les ressources humaines du projet et les tâches que ces personnes ont accomplies. La section 6 discute de la planification du projet.

2 Contexte

Au début de l'année 2003, une phase d'analyse a été menée afin de déterminer comment organiser le travail de la manière la plus optimale en tenant compte des ressources disponibles et du temps alloué au projet. Certains des éléments listés ci-après furent discutés lors du rapport d'étape #1. Il apparaît néanmoins pertinent de présenter à nouveau certains de ces éléments afin de faire bien illustrer le contexte particulier dans lequel se fait le développement de Modeleur2.

Travailler avec une petite équipe

Étant donné les ressources financières limitées dans le cadre du projet SEGRI, le développement se fait avec une équipe réduite. Le cœur de l'équipe est constitué de deux développeurs seniors auxquels se greffent des stagiaires (le nombre varie entre un et trois). Pour compenser la diminution de la force de travail en comparaison avec Modeleur 1.0 (environ six à huit personnes), il a donc fallu revisiter plusieurs des aspects du développement logiciel afin d'être en mesure de construire le logiciel, avec un niveau de qualité élevé et un jeu de fonctionnalités qui répond aux besoins du projet et aux attentes des partenaires et des utilisateurs.

Développer avec des stagiaires

Pour que le projet avance au rythme désiré, il est impératif que les stagiaires qui viennent travailler au sein du groupe laissent une contribution tangible et réutilisable. Le mot « réutilisable » est important ici car il faut bien voir qu'après le départ du stagiaire, il doit être aisé de reprendre le travail là où il l'a laissé et de continuer à avancer de manière quasi-transparente.

Un stagiaire demeure typiquement environ quatre mois au sein du groupe. Il a donc fallu mettre en place une structure adaptée à ce très haut taux de roulement. Dans cette optique, les mandats de stage sont découpés en tranches d'environ quatre mois. Pendant cette période, le stagiaire se voit confier un dossier complet (analyse, design, programmation, test) pour lequel il se voit remettre un échéancier. Il agit alors en tant que pilote. C'est à lui qu'il revient d'organiser son travail et de tenir des réunions d'analyse au besoin.

À chacune de ces réunions, il doit produire un rapport. Ces rapports de réunion sont importants car ils permettent de garder une trace des décisions qui ont été prises tout au long du dossier, et les raisons pour lesquelles elles ont été prises. À la fin de son stage, le stagiaire doit rédiger un rapport détaillant chacune des tâches effectuées dans le cadre de son dossier. Ce document sert par la suite de référence une fois que le stagiaire a quitté.

Construire autour d'un bus logiciel

L'architecture est centrée autour d'un bus logiciel auquel sont connectés des modules. Les modules sont des pièces logicielles capables d'envoyer des événements sur le bus et d'en recevoir de celui-ci. Cette approche bus / modules a été retenue car elle minimise le couplage entre les modules eux-mêmes.

Il est donc possible d'en enlever, d'en ajouter, d'en interchanger sans qu'il y aie de répercussions sur le système en entier. Étant donné que tous les événements circulent sur le bus, il est possible d'enregistrer des macros en étudiant ce flot d'événements pour pouvoir le répéter par la suite.

L'approche « par module » permet de circonscrire clairement le rôle à jouer par chacun des modules, tout en gardant le minimum de dépendances avec les autres modules. Elle permet le développement de plusieurs modules de manière concurrente une fois l'interface « événementielle » déterminée. Ceci est très approprié à la définition des mandats de stage, car les liens de dépendance entre les tâches sont minimes. Il est également plus aisé de corriger les bogues trouvés dans un module sans avoir à toucher au reste du logiciel.

S'assurer une base fonctionnelle en tout temps

Le développement logiciel est souvent ralenti par l'étape d'intégration. L'approche bus / modules permet d'atténuer en partie les effets de cette étape.

L'autre moyen qui est mis en œuvre est le suivant: s'assurer de toujours avoir une base de code solide dans le gestionnaire de code source CSV (Concurrent Version System), base avec laquelle tous les développeurs travaillent. Les développeurs ajoutent leur contribution à cette base de code, mais à tout moment durant le développement il est possible de construire un version du logiciel fonctionnelle à partir les sources dans CVS.

Dans cette optique, un système automatisé de compilation est en place. La nuit, le système en question extrait de CVS les modules nécessaires à la construction du logiciel et les reconstruit pour une gamme élargie de compilateurs. Il rend disponible les résultats de compilation sur le site intranet du groupe.

S'appuyer sur des composantes externes

Vu la gamme d'améliorations et d'ajouts à faire à Modeleur dans le cadre du projet SEGRI, il a fallu chercher un moyen de diminuer la banque de code à la charge du groupe.

Le projet repose donc, dans la mesure du possible, sur des librairies externes gratuites et disponibles sur Internet sous des licences « open source ». Dans cet optique, l'engin de base de données, le système de visualisation et le langage de script sont issus de composants externes.

S'assurer d'être extensible pour l'avenir

Construire un outil qui comblera les besoins de tous et chacun n'est pas une tâche facile, surtout dans le cas d'un logiciel scientifique complexe comme l'est Modeleur. Pour répondre à cette problématique, il a été décidé que l'outil ne doit pas nécessairement offrir directement des fonctionnalités permettant de faire tout ce qui est imaginable. Il doit plutôt fournir une structure flexible pour pouvoir le faire.

La distinction est importante car elle signifie que ce qui sera livré en fin de projet est une base de logicielle solide et fonctionnelle qui couvrira les besoins de la plupart des usagers. À celle-ci, il sera possible d'intégrer des ajouts externes afin d'augmenter les capacités du logiciel et d'ainsi augmenter sa vie utile. L'important est que les mécanismes sont en place pour que le logiciel réponde aux besoins des usagers pour longtemps, en leur donnant la possibilité d'ajouter des fonctionnalités additionnelles (via des plug-in).

Rendre disponible le logiciel

Il été déterminé que la structure des licences telle qu'elle l'est actuellement pourrait être revue. En effet, puisque le logiciel est construit d'éléments qui font partie de la communauté « open source », il est possible que Modeleur2 soit lui aussi éventuellement intégré à cette communauté.

Dans cette optique, Modeleur2 est distribué à l'aide d'un système similaire à Cygwin (émulation de la plateforme Unix sous Windows). Ce système permet la distribution et la mise à jour des différentes composantes de Modeleur2 à partir du web.

L'intérêt d'utiliser un tel système est de faire disparaître à toute fin pratique l'étape de distribution du logiciel. Si éventuellement des plug-in sont développés avec des intérêts commerciaux, un système de licences pourrait être mis en place pour combler ce besoin.

3 Spécifications fonctionnelles

Afin de produire un logiciel qui répond aux besoins des utilisateurs, il importe de rencontrer ces utilisateurs et d'aller chercher leurs commentaires et suggestions.

Dans le cas de Modeleur2, cette étape avait été réalisée en 2003 au moyen « d'entrevues » avec les utilisateurs / partenaires. C'est ainsi qu'avaient été rencontrées les personnes suivantes :

- José Bechara (Université Nationale du Nord-Est, Corrientes, Argentine) ;
- Jean Morin, Olivier Champoux, Daniel Rioux (Environnement Canada);
- Paul Boudreau, Michel Leclerc (INRS-ETE);
- Renaud le Boulleur de Courlon (Institut National Polytechnique de Toulouse);
- Jean Gauthier (BPR);
- Richard Frenette (Université d'Ottawa) ;
- Pierre Bélanger (GPR);
- André Robitaille, Michel Carreau (Synexus Global).

Les spécifications fonctionnelles incorporent également toute l'expérience de l'INRS-ETE tant dans l'utilisation de Modeleur que dans la formation des utilisateurs de Modeleur / Hydrosim.

Une version préliminaire des spécifications fonctionnelles avait été rédigée en 2003. La version finale des spécifications fonctionnelles fut complétée en 2004 et est disponible dans le document Spécifications – Modeleur2.

4 Progression des travaux – année 2004

Avant de détailler la progression des travaux, il est important de mentionner que ce document reprend les grandes lignes des dossiers qui ont été abordés au cours de l'année 2004. Un rapport scientifique a été rédigé pour la plupart de ces dossiers. Ces rapports scientifiques sont disponibles dans le document Rapports de recherche 2004.

Janvier 2004

Arrivée de nouveaux stagiaires au sein du groupe

Le début de l'année 2004 a salué l'arrivée de nouveaux stagiaires au sein de l'équipe de développement. Maude Giasson, étudiante déjà diplômée en mathématiques de l'Université Laval, est venue compléter son certificat en informatique par un stage chez nous. Mme Giasson a d'ailleurs effectué un second stage au sein du groupe au cours de l'été 2004, pour enfin débuter une maîtrise en sciences de l'eau à l'INRS-Eau, Terre et Environnement à l'automne 2004. Mme Giasson allait se voir confier le mandat traitant du module Partition de maillage.

Cédric Caron, étudiant en génie informatique 4ème année à l'Université de Sherbrooke, est également venu s'intégrer au groupe. M. Caron allait se voir confier le mandat traitant du module MNT.

Benjamin Behaghel, étudiant finissant en génie informatique de l'école EPITA en France, est venu compléter le groupe de stagiaires pour le début de l'année 2004. M. Behaghel allait se voir confier le mandat de réaliser le module validation et de filtrage des données.

Finalisation des livrables 2003

Au tout début de l'année 2004, les livrables 2003 suivants furent révisés suite aux commentaires des différents intervenants:

- rapport d'étape #1;
- spécifications fonctionnelles – version préliminaire.

Une version finale de ces documents a été livrée à la fin du mois de janvier 2004. Ce livrable a servi pour obtenir l'approbation de financement pour la seconde tranche du projet.

Analyse de la base de données

Au tout début de l'année 2004, une analyse a été faite en ce qui a trait à la base de données afin d'améliorer les performances obtenues. La base de données utilisée en début d'année était PostgreSQL 7.3 (la même qui a été utilisée en 2003).

L'objectif était de pousser plus loin ce qui avait été réalisé en 2003 et de trouver des moyens d'optimiser le rendement. Les sujets suivants furent étudiés au moyen de prototypes:

- les vues BD (matérialisées et non matérialisées);
- la librairie DTL (http://dtemplatelib.sourceforge.net/index.htm);
- le langage PG/PLSQL;
- la commande COPY de PostgreSQL.

Contact avec des experts en base de données chez Environnement Canada

Suite à des performances en deçà des attentes pour le chargement / la sauvegarde d'un maillage, André Plante et Sylvain Martin, des experts en base de données chez Environnement Canada furent consultés. Un livrable a donc été préparé à leur intention afin de les introduire au contexte et à leur fournir le plus d'éléments susceptibles de les aider à intervenir efficacement.

Gestion des champs et des séries dans la base de données

Les sections de code relatives aux champs et aux séries avaient été développées en 2003. La gestion des champs et des séries au niveau de la base de données n'était toutefois pas implantée. Cette tâche fut réalisée au début 2004. Il est donc maintenant possible de charger et d'enregistrer des champs et des séries dans la base de données. Des tables ont été créées à cet effet dans la base de données et du code C++ supportant ces nouvelles tables a été produit.

Analyse pour supporter les isosurfaces et naissance de SRChampVector

L'objectif de cette tâche était de dissocier de sa représentation graphique l'objet isosurface en tant que résultat de calcul. Ainsi, certaines classes de Modeleur 1.0 ont été récupérées et ajustées. L'analyse a amené la création de la structure de données SRChampVector.

Ajustements au BDTraducteur pour supporter les semis de points

Le BDTraducteur qui avait été développé en 2003 fut mis à jour pour supporter la traduction des semis de points provenant de Modeleur 1.0.

Écriture d'un script pour installer PostgreSQL 7.4

L'installation de PostgreSQL (http://www.postgresql.org/) et PostGIS (http://postgis.refractions.net/) nécessite de compiler et de construire plusieurs librairies. Jusque là cette tâche devait être effectuée dans un environnement Cygwin (http://www.cygwin.com/) une émulation de l'environnement Unix sous Windows. L'objectif de la tâche était de ne plus exiger la présence de Cygwin sur la machine de l'utilisateur. Pour ce faire, il a fallu rendre possible l'installation de l'engin de base de données via un script d'installation responsable d'effectuer toutes les opérations nécessaires.

Révision des normes de programmation

Certaines des normes de programmation en vigueur au sein du groupe ont été revues et améliorées. Un nouveau document a été mis en circulation à l'intérieur du groupe.

Simplification du module d'affichage

Le module d'affichage a été revu et simplifié. Le design fait en 2003 était basé sur structure souple mais qui était complexe à maintenir en pratique. Celle-ci a donc été simplifiée afin de ne supporter que l'engin VTK (Visual ToolKit - http://public.kitware.com/VTK/). Si d'autres engins graphiques viennent à être supportés, certaines sections de code devront être ajustées pour permettre l'utilisation d'algorithmes de tracés autres que ceux VTK.

Ajustement des algorithmes de tracé pour les nœuds et le maillage

Les algorithmes de tracé pour les nœuds et le maillage ont été retravaillés pour être conformes au nouveau design. Certains liens avec des structures Modeleur 1.0 ont également été retirés (ex : la liste d'affichage).

Remise en marche l'année 2004 sur le plan matériel et logiciel

Le plan de backup a été mis à jour pour l'année 2004. Les cassettes de back up de l'année 2003 ont été remisées. Les besoins matériels ont été réévalués. Le logiciel Visual Studio .NET 2003 a été commandé. Les différents outils de développement FastBugTrack (http://www.fastbugtrack.com/), Visual Paradigm (http://www.visual-paradigm.com/) et Case Studio 2 (http://www.casestudio.com) ont été mis à jour sur les stations des développeurs.

Février 2004

Affichage de points sous VTK

Un prototype a été développé pour afficher les points à l'aide de VTK. Dans le processus, divers aspects ont été étudiés et répertoriés dans un document détaillant les connaissances recueillies sur la librairie VTK.

Recrutement d'un stagiaire technicien

Dave Guérin, un stagiaire du Cégep Lévis-Lauzon en informatique industrielle fut recruté pour un stage de 8 semaines. Son mandat allait être de rendre le code développé jusqu'ici compatible avec une série de compilateurs. Son intervention a permis de solidifier le processus de compilation automatisée.

Analyse pour le module MNT

Le module MNT a pour objectif de permettre à l'usager de rassembler toutes les partitions décrivant son terrain dans un même container. Ce container pourra servir à définir une zone qui correspond à la plus petite zone commune à toutes les partitions, et être utilisé pour définir la zone de couverture du maillage. La première tâche a été d'identifier les grands acteurs impliqués dans cette activité. Ceci a fait ressortir le besoin pour un acteur de type « éditeur de structures géométriques » .

Analyse du module affichage : caméra, picking et communication avec les modules

Des diagrammes de séquences furent dessinés pour faire ressortir toutes les tâches que le module d'affichage doit prendre en charge, notamment en ce qui a trait à la gestion de la caméra, au picking et à la communication avec les autres modules. Ces diagrammes ont servi de point de départ à la tâche module Éditeur.

Analyse du module éditeur

Au moins deux modules ont fait ressortir un besoin pour un éditeur de structures géométriques. Il était donc logique de chercher à combiner les besoins des deux intervenants et d'ainsi créer un éditeur générique et réutilisable. Une analyse a été menée en ce sens.

Design et implantation du SRChampVector

L'objet SRChampVector est conçu pour supporter des géométries (polygones, polylignes, points) et de porter des valeurs sur les points de la géométrie. Le design et l'implantation de cet objet furent réalisés. Le SRChampVector était alors lié à la librairie Terralib (http://www.terralib.org/), mais ceci fut modifié en fin d'année.

Mise en place de la notion de projection cartographique GIS

Toutes les données manipulées sous Modeleur2 sont dans la projection cartographique du projet. Cette projection est spécifiée par l'usager, mais on lui ajoute un décalage pour ramener les données à des valeurs où on évite les pertes de précision. Chacune des données extraite de la base de données est donc traduite de sa projection GIS d'origine vers celle du projet. Symétriquement, chacune des données est écrite dans la base de données en la traduisant de la projection projet vers celle de la donnée. Cette tâche a nécessité des ajustements dans chacun des algorithmes de lecture / écriture dans la base de données, de même que l'ajout de tables dans la base de données.

Parallélisation et stratégie pour le partitionnement du maillage

En vue de supporter la parallélisation, une stratégie de partitionnement du maillage a dû être développée. Elle vise assigner à chaque processus une partie du maillage et de gérer les échanges d'information nécessaires entre les processus. Les structures internes doivent travailler dans une numérotation locale mais être capable de se positionner dans le maillage global. L'objectif est de permettre à chaque processus d'effectuer un travail local, mais qu'il soit possible d'ensuite recombiner le travail fait par chacun, comme si le travail avait été fait sur un maillage global. Maintenant que les structures internes sont prêtes, il devient possible d'utiliser un partitionneur comme ParMETIS (http://www-users.cs.umn.edu/~karypis/metis/parmetis/) et des libraires de calcul parallèle comme PETSc (http://www-unix.mcs.anl.gov/petsc/petsc-2/).

Mars 2004

Design et implantation du module éditeur

Le design du module éditeur a été effectué au moyen de diagrammes de séquences et de diagrammes de classes. Une fois cette étape complétée, l'implantation eut lieu. L'éditeur permet l'édition de structures géométriques à l'aide de l'engin graphique VTK. Les opérations de création / effacement de points, de polylignes, de polygones sont supportées. Les opérations « scinder une ligne » et « fusionner deux polygones adjacents » sont également permises.

Implantation de l'interface d'édition pour le container SRChampVector

L'éditeur permet l'édition de containers de géométries via une interface (API). Le container de géométries peut donc être de n'importe quel type du moment qu'il supporte les opérations prescrites par l'interface. Le premier container à éditer fut le SRChampVector. L'interface fut donc implémentée sur celui-ci.

Écriture d'un document pour migrer de RCS vers CVS

Les fichiers de Modeleur 1.0 étaient anciennement sous l'outil de gestion de code source RCS. Chaque fichier possède l'historique des modifications qui ont été effectuées sur lui-même. Dans l'importation du fichier sous CVS, l'historique des modifications ne devait pas être perdue. Un document a été rédigé pour détailler la procédure à suivre pour compléter cette opération.

Remise en fonction d'un outil pour les tests unitaires

Un outil développé dans le cadre de Modeleur 1.0 (GenTest) fut remis en marche et utilisé pour effectuer les tests unitaires de certaines classes sous Modeleur2.

Définition d'un API pour les mailleurs

Il était souhaité de pouvoir intégrer de nouveaux mailleurs à Modeleur2. Pour cela, une interface commune à tous les mailleurs a dû être définie. Tout mailleur pourra être rattaché à Modeleur2 en respectant l'interface prescrite.

Design et implantation du module MNT

Les clases du module MNT ont été définies au moyen de diagrammes. Le modèle élaboré a ensuite été codé. Il s'agissait de la première ébauche du MNT.

Analyse – critères et « filtre » de sélection

Une analyse a été menée pour rendre disponible aux modules un moyen de « filtrer » les données par critères (ex : la donnée dépend du maillage x et est de type champ de topographie).

Protocole d'échange de données

Afin de pouvoir exécuter des calculs répartis sur différentes machines, il doit pouvoir y avoir communication entre les processus. En effet, un processus doit parfois communiquer avec un autre afin de requérir les voisins. La synchronisation entre les processus et l'établissement d'un protocole d'échange de données sont donc des éléments qui ont été mis en place afin d'accomplir cette tâche. La libraire externe MPI (http://www-unix.mcs.anl.gov/mpi/mpich/) fut utilisée pour la compléter.

Avril 2004

Tests effectués sur l'éditeur

Une batterie de tests fut développée pour tester le module éditeur. Ce banc de tests automatisé valide la plupart des fonctionnalités offertes par l'éditeur.

Rapport sur le module Éditeur

Un rapport sur l'éditeur fut rédigé. Le rapport explique les décisions qui furent prises dans ce dossier. Il présente également les choix de design qui ont été faits pour implanter le système et définir l'interface.

Mise en place de mécanismes pour recevoir des événements en Python

Un premier module a nécessité de recevoir des événements en Python. Une analyse a donc été faite pour supporter un mécanismes de « listeners » et de « callback », similaire à ce qui est fait dans le langage Java.

Mise en place de fonctionnalités de gestion de fenêtres

La gestion des fenêtres a été améliorée. Les modules ont la possibilité de créer et de détruire des fenêtres en utilisant les fonctionnalités de la fenêtre principale.

Spécifications pour l'interaction usager vs Modeleur2

Différentes discussions ont eu lieu afin de définir un modèle d'application cohérent et intuitif pour l'usager. Celles-ci ont permis d'élaborer une ébauche de prototype basée sur deux zones de contrôle et une zone fenêtres.

Analyse du module Partition de maillage

Les grands intervenants du module Partition de maillage furent ciblés. Une réutilisation du code fait dans le module MNT fut envisagé.

Bilan des tâches restantes

Un bilan de toutes les tâches connues à ce jour fut dressé. Ce bilan allait servir par la suite à dresser un plan de développement pour la période estivale.

Arrivée d'un stagiaire pour travailler sur la modélisation de température

Daniel Nadeau, stagiaire en génie physique 3ème année issu de l'Université Laval, s'est joint à l'équipe. Son mandat allait être d'effectuer des simulations de températures avec le simulateur H2D2 développé dans le cadre de Modeleur2.

Migration de code dans H2D2

Le code permettant les calculs de convection/diffusion de température, ceux de convection/diffusion de contaminant conservatif et ceux de convection/diffusion de coliformes fécaux fut migré. Le travail de réécriture des équations de St-Venant pour faciliter l'intégration avec le modèle 2D ½ a été commencé, mais n'est pas complété au moment d'écrire ces lignes.

Mai 2004

Étude de GEOS

La librairie Terralib qui fut choisie en 2003 ne supportait pas certaines opérations géométriques rendues nécessaires pour les tâches des modules MNT et Partition de maillage. Afin de combler cette lacune, la librairie GEOS (http://geos.refractions.net/) elle aussi une librairie GIS géométrique, fut étudiée en vue de remplacer Terralib.

Migration vers des séquences pour être compatible avec Oracle

Suite aux commentaires reçus d'Environnement Canada, le code a été revisité en entier pour supporter la notion de séquences, changement nécessaire pour être compatible avec l'engin de base de données Oracle. Lors de cette opération, le type de donné utilisé dans le code a été changé pour supporter les entiers 64 bits. Ces changements ont touché autant le code Modeleur2 que le code du traducteur.

Mise en place des objets graphiques et d'une librairie de traçage VTK

Les couches graphiques se sont vues renommées par le terme « objet graphique ». Un objet graphique est en général lié à une donnée et possède une méthode pour synchroniser sa représentation graphique avec celle-ci. Une librairie de traçage VTK a été mise en place pour regrouper des classes de tracés. Les objets graphiques utilisent dans leur méthode de synchronisation les classes d'algorithmes de tracé de la librairie VTK pour générer la représentation graphique.

Séparation entre le module visualisation et le module affichage

Le module affichage a été scindé en deux parties. Une distinction claire a été faite en ce qui a trait au rôle de chacun des deux modules. Le module affichage est responsable des tâches d'affichage des objets graphiques alors que le module visualisation est responsable de construire des objets, de les remplir et de les envoyer au module d'affichage.

Définition de la structure de la base de données pour le MNT / PM

Le design des tables de la base de données pour supporter les composants du MNT et des Partitions de maillage a été réalisé.

Choix d'être toujours couplé à la base de données pour les semis de points

En raison de la grosseur des jeux de données, il est impossible ou encore peu souhaitable d'avoir toutes les données en mémoire. Pour cette raison, un système par tuiles a été développé. Les tuiles sont déchargées au besoin, et dans l'opération, tout changement à l'un des points est également écrit dans la base de données.

Design du module Partition de maillage

Le design du module de partition de maillage a été fait en s'inspirant du MNT. Le module a été développé en se basant sur le « design pattern » du proxy pour ajouter et configurer les mailleurs.

Design du module de filtrage

Le design du module de filtrage a été réalisé. La concept de filtre primaire et secondaire a été pensé pour permettre les combinaisons de filtres. Il a été déterminé que tout comme la partition de maillage, un système basé sur des « proxy » était souhaitable pour supporter l'ajout de filtres externes issus de DLL ou bien de scripts Python. Une interface a été définie pour supporter l'ajout de filtres externes.

Discussions pour l'organisation des menus

Des discussions ont eu lieu et une ébauche de l'arborescence des menus a été faite. Bien qu'il y ait eut un consensus sur l'idée de base, l'organisation finale des menus a été repoussée à une étape ultérieure, plus près du moment du relâchement de la première version de Modeleur2.

Présentation sur l'utilisation d'outils de publication Internet

Suite à une présentation à l'Université Laval sur l'utilisation d'outils de publication Internet, il été envisagé d'améliorer le site intranet du groupe pour rendre disponible la banque de connaissances à tous (rapports, liens Internet, etc.).

Mise à l'épreuve de H2D2

Le logiciel H2D2 fut utilisé dans la cadre d'un problème de modélisation de températures. Ceci a permis de corriger des problèmes et du même coup de valider que l'aspect scientifique de la modélisation de températures était implanté correctement.

Horaire de développement avec MS Project™

Le plan de développement a été révisé et ajusté. Le modèle Microsoft Project™ fut remis à jour suite au travail effectué durant les quatre premier mois. Les tâches restantes et les durées furent réévaluées pour définir une plan de développement pour les mois suivants.

Prototype – gestion des barres d'outil

Le prototype s'est vu ajouté une composante pour la gestion des barres d'outils. Tout plug-in peut ainsi proposer une barre d'outil à Modeleur2 et l'ajouter dans la zone réservée à cet effet.

Rencontre avec Bernard Doyon (Environnement Canada)

Dans le cadre de la révision du plan de gestion des débit du fleuve pour le comité mixte international (CMI), une réunion fut tenue avec Bernard Doyon d'Environnement Canada.

Campagne de terrain sur la Rivière Chaudière

Une campagne de terrain sur la Rivière Chaudière a été menée dans le but de recueillir les données requises pour monter un Modèle Numérique de Terrain (MNT).

Juin 2004

Implantation de l'algorithme de maillage

Une première ébauche d'un algorithme de maillage a été implantée. Il était alors possible de mailler les couches d'une partition. Un mailleur de Modeleur2 a été migré et utilisé pour compléter cette tâche.

Gestionnaire de Partition de maillage

Le gestionnaire de partition de maillage a été implanté. Il répond aux événements de création et d'ouverture de partitions de maillage et de couches de maillage.

Migration de Terralib vers GEOS

La décision a été prise d'arrêter d'utiliser la librairie Terralib pour gérer les géométries à l'aide de structures propriétaires à Modeleur2. C'est seulement au moment de faire une opération géométrique que des structures GEOS seront construites pour faire le travail (ex : intersection, union, différence, etc).

Diagramme de séquences pour le module projet

Les diagrammes de séquences pour le module projet ont été réalisés. Ils ont permis de cibler les grands intervenants et de définir la communication entre eux.

Supporter les schémas et les projets

Afin de mettre en place la notion de projet au niveau du logiciel, les schémas ont été mis en place dans la base de données. Le traducteur a dû être modifié en conséquence.

Mise en place de liens vers les données externes au projet

Afin de permettre l'utilisation de données issues d'autres projets, la notion de lien externe a été mise en place. Les liens externes sont pour le moment uniquement permis sur des projets de la même base de données

Rapport de développement sur le module de validation

Un rapport résumant les principaux sujets reliés au dossier module de validation des données a été rédigé.

Rapport de développement sur le module de filtrage

Un rapport résumant les principaux sujets reliés au dossier module de filtrage des données a été rédigé.

Recrutement de stagiaires pour l'automne 2004

Sébastien Labbé, étudiant en mathématiques 3ème année à l'Université de Sherbrooke, fut recruté pour travailler sur le logiciel. M. Labbé allait avoir pour tâche de compléter et améliorer les modules Partition de maillage et MNT.

Vincent Martineau, étudiant en informatique 3ème année à l'Université Laval, a lui aussi été recruté. Son mandat était de participer à l'amélioration du logiciel en vue du relâchement d'une version initiale prévue à la fin décembre 2004.

Rédaction d'un guide de publication en Python

Afin de pouvoir utiliser les classes C++ dans les scripts Python, celles-ci doivent être publiées. Cette tâche doit être faite régulièrement afin de rendre disponible les créations C++ en Python. Afin de faciliter cette tâche et de l'uniformiser pour tous les développeurs, un guide de publication de C++ vers Python a été rédigé.

Développement à distance

L'environnement de développement de Modeleur2 a été monté de façon à ce qu'il soit possible d'effectuer du développement à distance. Ceci fut validé lors d'essais concluant au Brésil. Le tout a été mis en place là bas avec succès. Du travail sur la base de code de Modeleur2 a ainsi pu être réalisé de la même façon qu'il l'aurait été au Québec.

Mise à jour du Modèle Numérique de Terrain (MNT) de la Rivière Chaudière

Le MNT initialement construit par Renaud le Boulleur de Courlon en 2003 fut mis à jour à l'aide des données recueillies sur la Rivière Chaudière lors de la campagne de terrain.

Juillet 2004

Mise en place des objets graphiques de type SRChampVector

Le code a été rédigé pour représenter visuellement des SRChampVector. Le code est utilisé pour représenter les contours des couches de maillage et des couches des partitions d'un MNT.

Mise en place d'un objet graphique composite

Afin d'héberger les objets graphiques de contour de chacune des couches, un objet graphique composé d'autres objets graphiques fut mis en place. Cet objet graphique se comporte exactement comme un objet graphique standard, mais renvoie simplement les appels aux objets graphiques qu'il contient.

Mise en place des méta données

La notion de métadonnée fut mise en place. Une classe pour les métadonnées fut définie, de même que des événements d'assignation et de requête des métadonnées.

Mise en place d'événements de liste d'entités

Des événements pour lister les entités d'un projet furent créés. Pour le moment, les requêtes de listes sont simples, mais dans l'avenir il sera possible d'effectuer des requêtes de liste en fonction de critères plus élaborés.

Modification des tables de la BD pour supporter l'héritage d'objets graphiques

La structure de la base de données a été revue pour supporter une hiérarchie au niveau des objets graphiques, similairement à celle qui fut implantée pour les tables des MNT et des partitions de maillage.

Mise en place de l'algorithme du gestionnaire de BD pour le « enregistrer sous »

Une distinction claire fut faite en ce qui a trait aux tâches de « enregistrer sous » et celles de « enregistrer ». La hiérarchie des algorithmes dans la base de données s'est donc vue inclure un algorithme pour le « enregistrer sous ».

Mise en place de mécanismes de transition éditeur / module

Une dynamique a été mise en place pour que l'objet graphique d'une couche qui est envoyée en édition soit mis invisible le temps de l'édition. C'est l'éditeur qui a le contrôle et qui modifie la structure géométrique qui lui a été passée (dans le cas des couches, c'est le SRChampVector).

Mise en place de la sélection multiple

La sélection multiple a été mise en place au niveau de l'arborescence de la partition de maillage. Il est donc possible de demander le maillage de plusieurs couches dans la même opération.

Août 2004

Mise en place d'échelles de couleurs

La gestion des couleurs a été mise en place. Les couches des partitions portent désormais une couleur. L'échelle de couleur peut être constante ou variable (ex : linéaire, logarithmique).

Rédaction d'un rapport sur le module partition de maillage

Un rapport résumant les principaux sujets reliés au dossier module de partition de maillage a été rédigé.

Migration du module MNT pour le rendre similaire au module Partition de maillage

Le module MNT et le module Partition de maillage sont très similaires dans leur comportement. Ils diffèrent toutefois du fait que le module MNT maintient une arborescence à trois niveaux et que le module Partition de maillage en maintienne une à deux niveaux.

Rédaction d'un rapport sur les fonctionnalités manquantes et les points à améliorer

Suite au développement effectué au cours des mois précédent, un rapport fut rédigé. Il y était question des fonctionnalités manquantes ainsi que des points à améliorer ou bogues à corriger. Ce rapport a servi de base à la rédaction du plan pour le livrable de fin d'année 2004.

Septembre 2004

Mise à jour de l'outil de bug-tracking

La liste des bugs a été remise à jour en fonction des bogues connus et des améliorations souhaitables pour la version du livrable fin 2004. Les bogues furent assignés aux membres du groupe et la liste constitue une base pour les tâches de suivi du projet.

Préparation d'un plan pour la version fin 2004

Une liste de fonctionnalités a été établie en vue de la publication de la première version du logiciel (livrable fin 2004). Cette liste a servi à la rédaction d'un plan de développement, lequel a été proposé aux membres du groupe.

Script de démarrage et ajout des plug-in

La façon d'ajouter des plug-in a été améliorée. Un script de démarrage de l'application a été mis en place. Les plug-in (sections C++ et sections Python) peuvent désormais être ajoutés de l'extérieur sans qu'il soit nécessaire de rouvrir le logiciel.

Refonte du gestionnaire de données

Les GDAlgoIo (les algorithmes d'écriture / lecture dans la base de données) ont été revus et améliorés. Auparavant, leur ajout était *hard codé* dans le gestionnaire de données. Maintenant, chaque plug-in a la responsabilité de faire connaître au gestionnaire de données les algorithmes qu'il rend disponible à tous les autres plug-in.

Déménagement

L'INRS a déménagé au centre-ville de Québec. Des préparatifs ont dû être faits pour préparer le déménagement. Une certaine période de temps fut nécessaire pour aménager dans les nouveaux locaux et remettre le groupe opérationnel.

Traçage des nœuds et des maillage amélioré

La librairie TraceVTK a été améliorée pour utiliser plus efficacement les fonctionnalités VTK. Suite à cela, les tracés des maillages et des nœuds (de maillage) ont été améliorés. Ils supportent désormais une couleur associée à un champ scalaire.

Entrée / sortie du mode édition

L'interaction des modules avec l'éditeur a été modifiée. Une action explicite de l'utilisateur est maintenant requise pour entrer et sortir du mode édition, alors qu'auparavant ceci était provoqué par un double-clic. L'action double-clic est réservée pour donner le focus à une fenêtre et ne doit donc pas amener celle-ci en édition.

Création de colonnes GIS

Des mécanismes ont été mis en œuvre pour permettre la création de colonne GIS dans la base de données en tenant compte du nom du schéma (projet actif). Le traducteur a également été modifié en conséquence.

Macros

Le code pour enregistrer et rejouer une macro a été rendu fonctionnel. Un petit exemple a été développé pour démontrer l'utilisation des macros. Il est donc possible de démarrer l'enregistrement, d'enregistrer une action de l'usager et de rejouer la macro.

Analyse mode référence

Une analyse a été réalisée en ce qui a trait à la logistique création / chargement/ déchargement au niveau du gestionnaire de données. L'analyse a été menée en cherchant la stratégie à adopter pour le transfert de propriété des données et les liens de dépendance.

Connecter les fonctionnalités de la partition de maillage avec la BD

Les fonctionnalités de la partition de maillage qui ont trait à la sauvegarde et au chargement de partitions de maillage dans la base de données ont été mises en place.

Mémoire de maîtrise de Laurent Bonnifait

Le mémoire de maîtrise de Laurent Bonnifait fut envoyé aux correcteurs. Michel Leclerc a agit en tant que directeur maîtrise pour M. Bonnifait, alors que M. Yves Secretan et Mme Monique Bernier ont a agi à titre de co-directeurs. Le mémoire est toujours en cours de révision au moment d'écrire ces lignes.

Octobre 2004

Supporter les zones d'étude

La fonctionnalité de zone d'étude fut mise en place. L'usager est maintenant en mesure de spécifier une zone d'étude au moment de charger toute donnée géoréférencée dans Modeleur2.

Décision mode référence / mode copie

Les spécifications fonctionnelles de Modeleur2 – version préliminaire avait été rédigées en ayant à l'esprit que les données allaient pouvoir être maintenues par copie ou par référence. La décision a été prise de supporter uniquement le mode par copie.

Refonte de l'arborescence MNT /PM

Le code qui traite la zone de contrôle (arborescence) des MNT / PM a été revu et amélioré. Les deux entités ont été comparées et un modèle fut mis en place afin de pouvoir réutiliser le code de l'un dans l'autre.

Travail sur les spécifications fonctionnelles

Les sections décrivant l'interface graphique et le gestionnaire de données furent rédigées. La section sur le module validation fut revue et améliorée en fonction des développements survenus en cours d'année sur ce dossier.

Intégration du mailleur Delaunay

Les mailleurs Frontal, Trivial et Delaunay furent intégrés à Modeleur2. Ces mailleur peuvent être utilisé pour générer des maillages. Les mailleurs furent migrés à partir du code de Modeleur 1.0.

Importation de semis de points

L'importation de semis de points avec le format GPR fut mise en place. Des mécanismes d'importation ont été implantés pour s'assurer que l'usager soit averti s'il y a un quelconque problème durant l'opération.

Modification dans la hiérarchie des champs

La hiérarchie des classes de champs fut revue et modifiée. La hiérarchie propose maintenant un héritage multiple. Les combinaisons (éléments finis / analytique) et (scalaire / vectoriel / ellipse erreur) peuvent maintenant être faite. Il y a donc au minimum six types de champs possible. Toutes les classes et tous les fichiers Python ont été revus durant le processus.

Ajout / modifications d'algorithmes BD pour gérer les champs

Les champs analytiques furent ajoutés au gestionnaire de données. La gestion des champs éléments-finis fut également revue pour implanter une notion d'héritage au niveau de la base de données de même que pour intégrer les champs ellipse erreur.

Recrutement

Une phase de recrutement fut tenue. L'objectif était de trouver un stagiaire pour la période de l'hiver 2005. Jean-Philippe Lemieux fut ainsi embauché. Il sera appelé à travailler sur le module de simulation.

Priorité des couches

La gestion des priorités des couches fut mise en place autant au niveau logique que visuel. Cette étape a provoqué la ré-ingénierie du module affichage.

Révision string OTL / TEXT

Les champs varchar qui étaient présents dans Modeleur2 furent remplacés par des champs TEXT dans l'outil de génération SQL. Les endroits où il y avait lecture de chaînes de caractères en OTL sont maintenant remplacés par des std ::string. Ce changement permet l'utilisation de chaînes de caractères de longueurs variables dans la base de données.

Ajustement des mailleurs

Les mailleurs ont été révisés pour supporter les opérations de maillage 1D (lignes) et de maillage 2D (zones) dans la même DLL.

Novembre 2004

Retirer couche / partition et fermer partition de maillage / MNT

Les opérations de fermeture de partitions de maillage et de MNT ont été implantées, de même que celles de retrait des éléments qui les composent (couches, partition). Le gestionnaire de données a été solidifié pour bien prendre en compte les opérations inhérentes au déchargement de ces structures.

Analyse sur la gestion des menus

Une analyse a été menée afin de bien cibler les besoins de chaque plug-in pour être en mesure de gérer efficacement les menus que chacun d'eux offre à l'utilisateur.

Analyse et design d'un nouvel algorithme de maillage

L'algorithme de maillage basé sur les priorités des couches fut amélioré. Des problèmes de précision géométrique avaient été rencontrés au cours de l'été avec le premier algorithme. Un second algorithme a donc été développé afin de pouvoir réaliser le travail sans avoir recours à des opérations géométriques.

Rédaction des spécifications fonctionnelles

Les sections décrivant le module de simulation, le module partition de conditions aux limites, le module visualisation, les outils calculatrice et construction de séries furent rédigés. La section sur le MNT et les partitions de maillage fut revue et améliorée en fonction des développements survenus en cours d'année sur ce dossier.

Implantation de la gestion des menus dans les plug-in

La gestion des menus fut implantée dans chacun des plug-in. Des événements furent créés pour permettre aux modules de prendre adéquatement la décision à savoir si chacun de leurs menus doit être désactivé ou non.

Ré-ingénierie du module affichage

Le module affichage fut revu et amélioré. Il répond maintenant à un seul événement et est en mesure de gérer adéquatement les acteurs VTK à l'aide de celui-ci. La gestion des espaces de représentation, des plans et des objets graphiques a été revisitée et améliorée dans le processus.

Objet graphique composite contenu dans une partition

Les partitions ont été modifiées pour inclure un objet graphique de type composite Le changement est applicable sur les MNT ainsi que sur les partitions de maillage.

Refonte du SRChampVector

Le SRChampVector fut modifié pour supporter les valeurs au niveau de chacun des ses points (à l'aide de template). Cela a permis d'alléger son interface, de le rendre plus facile à utiliser et prêt pour les besoins futurs. Dans l'exercice, certaines des géométries furent revues et modifiées.

Création et chargement de champ analytique en Python

Afin de mettre en place la notion de maillage adaptatif, les opérations sur les champs analytiques ont été publiées en Python. Il est maintenant possible d'assigner ce type de champ aux couches de maillage.

Boîte de dialogue pour la sélection de projection GIS

Une boîte de dialogue permettant la sélection de projection GIS fut développée. La boîte de dialogue requiert la liste des projections disponibles et permet à l'usager de choisir celle qui lui convient.

Installation de PostgreSQL 8.0

Le système de base de données PostgreSQL 8.0 fut rendu disponible en fin d'année. Pour la première fois, un installateur Windows était alors livré avec le logiciel. Ceci simplifie de beaucoup la tâche qui était auparavant relativement ardue. Un petit guide d'installation fut néanmoins publié à l'interne, lequel identifie les quelques étapes à suivre pour installer PostgreSQL 8.0 et être en mesure de l'utiliser.

Mise en place de la zone d'affichage par défaut d'une fenêtre graphique

La zone d'affichage par défaut d'une fenêtre graphique a été mise en place. L'usager peut désormais spécifier dans ses paramètres projet une zone qu'il désire voir afficher à l'ouverture d'une fenêtre graphique. La zone en question tient compte de la projection GIS. Jusque là, la zone d'affichage n'était pas prise en compte, et les modules redimensionnait la zone en fonction de ce qu'ils avaient à afficher.

Mise en place de fonctionnalités pour le zoom / translation interactif

La fonctionnalité de zoom / pan interactif a été implantée. L'usager peut désormais interagir avec la caméra intuitivement avec l'aide de la souris.

Publication du semi de points

La fonctionnalité permettant de publier un semi de points en champ de topographie a été mise en place. Cette opération sera exécutée par l'usager une fois qu'il a validé son semi manuellement ou à l'aide de filtres et qu'il désire le rendre disponible pour l'étape de construction d'un MNT.

Enregistrement des maillages générées à partir de la partition de maillage

La fonctionnalité permettant d'assembler ensemble tous les petits maillages d'une partition de maillage a été mise en place. Le maillage ainsi construit est continu et peut être utilisé pour effectuer des simulations.

Gestionnaire de zones de contrôle et de la zone de barre d'outils

La zone de contrôle est l'endroit où les plug-in attachent les éléments de l'interface graphique qu'ils publient (ex : arborescence MNT). La gestion de la zone de contrôle de Modeleur2 fut implantée pour permettre aux plug-in d'y ajouter du contenu. Dans la même veine, la zone d'ajout des barres d'outils fut elle aussi mise en place.

Décembre 2004

Assemblage des partitions de topographie sur le maillage

La fonctionnalité permettant l'assemblage de partitions de topographie chacune associée à un champ de topographie a été rendue disponible. Une boîte de dialogue permet à l'usager de choisir sur quel maillage il désire porter sa ou ses partitions.

Intégration des filtres

Le module de filtre développé à l'été n'était pas connecté avec les semis de points de Modeleur2. Cette fonctionnalité a été mise en place, et il est désormais possible de filtrer un semi de points, et d'obtenir les cotes associées à chacun des points. Les points filtrés sont alors affichés d'une couleur différente des autres pour qu'il soit possible de les différencier.

Boîtes de dialogue et tracé de points, maillage, vecteurs

Des boîtes de dialogue ont été développées pour permettre à l'usager d'effectuer toute une gamme de tracés (maillage, points, isolignes).

Éditeur de champ analytique

Un éditeur de champ analytique (texte) a été rendu disponible. Il est ainsi possible de créer, charger, sauver et sauver sous des champs analytiques. Cet éditeur permet de construire et éditer les champs analytiques associés aux couches de maillage.

La fonctionnalité Sauve sous

Une boîte de dialogue commune a été rendue disponible pour effectuer la tâche de sauvegarde sous. Tous les plug-in qui requièrent la sauvegarde d'une donnée utilisent désormais cette nouvelle boîte de dialogue.

Rédaction du rapport d'étape #2

Le rapport d'étape #2 a été rédigé. Celui-ci inclut une section sur les spécifications fonctionnelles, une section sur les tâches accomplies à chaque mois de l'année 2004, une section sur les ressources humaines et une section sur la planification du projet.

Supporter les métadonnées au niveau des champs

Les métadonnées ont été ajoutées aux entités de type champ. Il est désormais possible de requérir la liste des entités de type champ, plutôt qu'uniquement celles de type série. Le traducteur a dû être ajusté en conséquence.

Supporter un log

Un système de log a été mis en place en Python. Les erreurs sont désormais affichées dans ce log en fonction du niveau de log avec lequel est configurée l'application. Par exemple, les message de type debug ne sont pas affichés en mode release.

Supporter un splash screen

Un splash screen a été ajouté au démarrage de Modeleur2.

Correction de bogues

En vue de livrer une première version de Modeleur2, une série de tests ont été effectués et différents bogues furent corrigés.

5 Ressources humaines

Les personnes suivantes ont été appelées à travailler sur le projet SEGRI au cours de l'année 2004. La contribution de chacune de ces personnes est détaillée dans les prochaine pages.

Yves Secretan

M. Secretan, professeur à l'INRS-ETE, est responsable du projet SEGRI - volet informatique. Il a participé à la planification du projet et au recrutement de stagiaires. Il a travaillé à l'encadrement de ceux-ci et a participé à toutes les réunions de design et d'analyse tenues au cours des stages. Il agit également comme directeur de maîtrise de Maude Giasson et Khalid Rahman.

En plus d'offrir un support aux stagiaires, M. Secretan a effectué diverses tâches de développement logiciel. Il a notamment développé H2D2, la version 2 des logiciels Hydrosim et Dispersim. Il a de plus travaillé sur la plupart des modules du logiciel afin de les améliorer ou de leur ajouter des fonctionnalités manquantes.

M. Secretan a travaillé à la rédaction des spécifications fonctionnelles, et à la révision de nombreux rapports ou documents.

Michel Leclerc

M. Leclerc, professeur à l'INRS-ETE, est responsable du projet SEGRI – volet application sur la rivière Chaudière. Il fut directeur de maîtrise de Laurent Bonnifait. Il a également révisé les spécifications fonctionnelles de Modeleur2. M. Leclerc a également soumis pour publication l'article « Cartographie du risque unitaire d'endommagement (CRUE) par inondations pour les résidences unifamiliales du Québec » rédigé par Blin, Leclerc, Secretan et Morse pour la Revue des Sciences de l'eau.

Paul Boudreau

M. Boudreau, agent de recherche, a révisé les spécifications fonctionnelles et fourni des suggestions d'amélioration du logiciel.

Eric Larouche

M. Larouche, ingénieur en informatique, a comme tâche d'agir en tant que développeur senior au sein du groupe. Il a participé à la planification du projet et au recrutement de stagiaires. Il a travaillé à l'encadrement de ceux-ci et a participé à toutes les réunions de design et d'analyse tenues au cours des stages.

En plus d'offrir un support aux stagiaires, M. Larouche a effectué diverses tâches de développement logiciel. Il a travaillé sur la plupart des modules du logiciel afin de les améliorer ou de leur ajouter des fonctionnalités manquantes

M. Larouche a travaillé à la rédaction des spécifications fonctionnelles. Il a également travaillé à la rédaction / révision de plusieurs documents d'analyse, documents de génie logiciel ou de rapports.

Maude Giasson

Mme Giasson, étudiante à l'Université Laval, a travaillé sur le projet SEGRI à titre de stagiaire au Certificat en informatique. Mme Giasson est également titulaire d'un baccalauréat en mathématiques. Son stage s'est effectué du 5 janvier au 13 août 2004.

Mme Giasson s'est vu confiée comme principal mandat de mettre en place le module partition de maillage et d'y implanter la fonctionnalité permettant de mailler des couches. Dans le cadre de son mandat, elle a eu à travailler de près avec les modules d'affichage et de visualisation.. Elle a également eu à mettre en place le SRChampVector, un objet container de géométries et porteur de valeurs.

Mme Giasson a débuté une maîtrise à l'automne 2004. Mme Giasson sera appelée à travailler sur des méthodes pouvant aider à la validation des données provenant de levées topographiques au laser aéroporté.

Benjamin Behaghel

M. Behaghel, étudiant à l'École Pour l'Informatique et les Techniques Avancées (EPITA, Paris, France), a travaillé sur le projet SEGRI à titre de stagiaire en génie informatique. Son stage s'est effectué du 12 janvier au 9 juillet 2004.

M Behaghel a travaillé sur le module de simulation et le module de filtrage des données. Il a également développé un éditeur de structures géométriques qui est réutilisé à plusieurs endroits dans le logiciel.

Kevin Solinski

M. Solinski, étudiant à Enic-TELECOM (Lille, France), a travaillé sur le projet SEGRI à titre de stagiaire en génie informatique. Son stage avait débuté en 2003 et s'est poursuivi jusqu'à la fin janvier 2004. Au cours de cette période il a été appelé à travailler à l'élaboration du site web de publication des résultats de compilation.

Cédric Caron

M. Caron, étudiant à l'Université de Sherbrooke, a travaillé sur le projet SEGRI à titre de stagiaire en génie informatique. Son stage s'est effectué du 5 janvier au 16 avril 2004.

M. Caron s'est vu confié comme principal mandat de mettre en place le module MNT. Il a également travaillé à la mise en place des algorithmes du gestionnaire de données pour effectuer la sauvegarde et le chargement des champs et des séries.

Daniel Nadeau

M. Nadeau, étudiant à l'Université Laval, a travaillé sur le projet SEGRI à titre de stagiaire en génie physique. Son stage s'est déroulé du 26 avril au 30 juillet 2004.

M. Nadeau s'est vu confié comme mandat de valider le modèle de températures avec le logiciel H2D2. Il a somme toute agit comme premier utilisateur officiel et a permis de tester les fonctionnalités de base du logiciel.

Stéphane Lévesque

M. Lévesque, étudiant à l'Université Laval, a travaillé sur le projet SEGRI à titre de stagiaire en informatique. Son stage s'est effectué du 10 mai au 13 août 2004.

M. Lévesque s'est vu confié un dossier visant à mettre en place une première phase du gestionnaire de données ainsi que la notion de projet. M. Lévesque a travaillé de près avec la base de données pour implanter les schémas, les requêtes de listes de données et les métadonnées. Il a également mis en place une série de boîtes de dialogue préliminaires.

Sébastien Labbé

M. Labbé, étudiant à l'Université de Sherbrooke, a travaillé sur le projet SEGRI à titre de stagiaire en mathématiques. Son stage s'est effectué du 7 septembre au 23 décembre 2004.

M. Labbé s'est vu confié comme principal mandat continuer le développement des module MNT et Partition de maillage. Il a également élaboré et mis en place un algorithme pour effectuer les tâches de maillage qui n'est pas sensible aux problèmes de précision géométrique.

Vincent Martineau

M. Martineau, étudiant à l'Université de Sherbrooke, a travaillé sur le projet SEGRI à titre de stagiaire en informatique. Son stage s'est effectué du 7 septembre au 23 décembre 2004.

M. Martineau s'est vu confié diverses tâches de développement visant à compléter certaines modules ou en améliorer d'autres.

Dave Guérin

M. Guérin, étudiant au Cégep de Lévis-Lauzon, a travaillé sur le projet SEGRI à titre de stagiaire technicien en informatique. Son stage s'est effectué du 29 mars au 14 mai 2004.

M. Guérin a travaillé au rodage du système de compilation automatisé. Il a éliminé toute une gamme d'erreurs et d'avertissements qui survenaient sur certains compilateurs.

Laurent Bonnifait

M. Bonnifait, étudiant à la maîtrise à l'INRS-ETE, a travaillé sur le projet SEGRI jusqu'au 9 juillet. Son implication dans le projet avait commencé au mois de juin 2003.

M. Bonnifait a effectué une campagne de terrain sur la Rivière Chaudière. Suite à cette campagne de terrain, il a intégré les données recueillies au Modèle Numérique de Terrain de la Rivière Chaudière. M. Bonnifait a également procédé à la rédaction de son mémoire de maîtrise intitulé « Développement de courbes submersion-dommages pour l'habitat résidentiel québécois ».

Khalid Rahman

M. Rahman a débuté une maîtrise à l'automne 2004. M. Rahman sera appelé à travailler sur le modèle de simulation 1D ½.

6 Planification

La section suivante présente le plan de développement pour toute la durée du projet SEGRI (2003-2005). Il couvre donc les deux ans écoulées jusqu'ici et une planification pour la dernière année. Dans un premier temps, les tâches-clés sont décrites brièvement. Dans un second temps, la planification du projet est présentée à l'aide d'un diagramme.

Les tâches sont découpées en tranches de quatre mois de manière à pouvoir être effectuées dans le cadre de stages. Les livrables logiciels sont prévus pour la fin 2004 et la fin 2005, et sont marqués à l'aide de jalons dans le diagramme.

6.1 Tâches

Certaines tâches sont plutôt associées à des tâches de maintenance. L'environnement de développement en est un exemple dans le sens où cette tâche est en constante évolution. Il est bien difficile de lui attribuer une valeur de pourcentage exacte. Elle est donc identifiée avec la mention 100% accompagnée du symbole '*'.

Environnement de développement

Cette tâche a pour but de mettre en place les mécanismes pour assurer le développement de Modeleur2 dans les règles de l'art du génie logiciel. Cette tâche inclut une composante rédaction de documents de génie logiciel (normes de programmation, analyses, rapports, recommandations d'outils). Elle inclut également la mise en place d'un système automatisé de compilation et de tests.

Cette tâche est complétée à environ 100%.*

H2D2

Cette tâche a pour but d'effectuer la migration d'Hydrosim / Dispersim. Cette tâche inclut également la parallélisation du logiciel afin de pouvoir distribuer la charge de calcul, et d'ainsi diminuer les temps de calcul.

Cette tâche est complétée à environ 75%.

Application Chaudière-MNT et maillage

Cette tâche a pour but de construire le Modèle Numérique de Terrain de la rivière Chaudière à l'aide de données provenant de différentes sources. Elle inclut également la conception d'un maillage adapté au cas de la rivière Chaudière.

Cette tâche est complétée à environ 60%.

Module de visualisation

Cette tâche a pour but de permettre la visualisation des maillages, des isolignes, des isosurfaces, des vecteurs, des semis de points, des nœuds, des profils en long et des lignes de courant.

Cette tâche est complétée à environ 80%..

Module de base de données

Cette tâche a pour but d'intégrer une base de données relationnelle à Modeleur2, et de fournir des mécanismes pour que les autres modules connectés au bus logiciel puissent créer, charger, sauver et détruire des données.

Cette tâche est complétée à environ 85%..

Champs-série

Cette tâche a pour but de définir et implanter les champs et les séries, structures de données qui seront à la base de Modeleur2.

Cette tâche est complétée à environ 95%.

Module de script

Cette tâche a pour but d'intégrer un langage interprété externe à Modeleur2 afin de remplacer la calculatrice, de permettre l'ajout de plug-in et de mettre en place un système de macro.

Cette tâche est complétée à environ 80%.

Spécifications fonctionnelles

Cette tâche a pour but de rencontrer les utilisateurs / partenaires et d'ensuite rédiger les spécifications fonctionnelles de Modeleur2.

Cette tâche est complétée à 100%.

Module GUI

Cette tâche a pour but d'intégrer à Modeleur2 les mécanismes permettant l'ajout de plug-in.

Cette tâche est complétée à 95%.

Interface graphique – boîte de dialogue

Cette tâche a pour but de développer les boîtes de dialogue présentes dans Modeleur2.

Cette tâche est complétée à 20%.

Module de partition de maillage

Cette tâche a pour but d'intégrer à Modeleur2 l'activité partition de maillage. La notion de maillage adaptatif, de même que la génération de transects pour les méthodes 1D sont inclus dans cette activité.

Cette tâche est complétée à 90%. Le rapport détaillant le travail effectué sur cette tâche est disponible dans le document Rapports de recherche 2004.

Module de modèle numérique de terrain (MNT)

Cette tâche a pour but d'intégrer à Modeleur2 une activité permettant de définir le Modèle Numérique de Terrain. Les partitions génériques sont incluses dans cette activité.

Cette tâche est complétée à 70%. Le rapport détaillant le travail effectué sur cette tâche est disponible dans le document Rapports de recherche 2004.

Module de validation des données

Cette tâche a pour but d'intégrer à Modeleur2 l'activité permettant d'effectuer la validation des données. Cette tâche inclut la mise en place de mécanismes permettant l'ajout d'algorithmes de filtrage externe.

Cette tâche est complétée à 70%. Le rapport détaillant le travail effectué sur cette tâche est disponible dans le document Rapports de recherche 2004.

Installation et distribution

Cette tâche a pour but de définir et de mettre en place les mécanismes afin de pouvoir effectuer la distribution du logiciel par le web.

Cette tâche est complétée à 75%.

Module de gestion de données

Cette tâche a pour but d'intégrer à Modeleur2 l'activité permettant de gérer l'import et export de données. Cette activité inclut également la gestion de projet et l'équivalent du gestionnaire d'entités de Modeleur 1.0.

Cette tâche est complétée à 60%. Le rapport détaillant le travail effectué sur cette tâche est disponible dans le document Rapports de recherche 2004.

Outils d'analyse

Cette tâche a pour but d'intégrer à Modeleur2 une gamme d'outils d'analyse. Parmi ceux-ci, on retrouvera des outils de ligne, de surface, de projection et de calcul de débit.

Module de simulation

Cette tâche a pour but d'intégrer à Modeleur2 une activité simulation. La connexion entre Modeleur2 et les simulateurs (H2D2, par exemple) sera faite à cette étape.

Aide et tutoriel

Cette tâche a pour but de définir et de mettre en place les mécanismes permettant de gérer l'aide et les tutoriels au niveau de chaque plug-in. Elle couvre également la rédaction du contenu de l'aide et des tutoriels.

Modèle 1 D ½

Cette tâche a pour but de mettre en place un modèle de simulation hydrodynamique 1D ½ dans H2D2.

Application Chaudière-Simulation

Cette tâche a pour but d'étudier le cas de la rivière Chaudière à l'aide de Modeleur2.

Évaluation des dommages

Cette tâche a pour but d'intégrer à Modeleur2 une composante « évaluation des dommages ».

Traitement des données raster, vectorielles

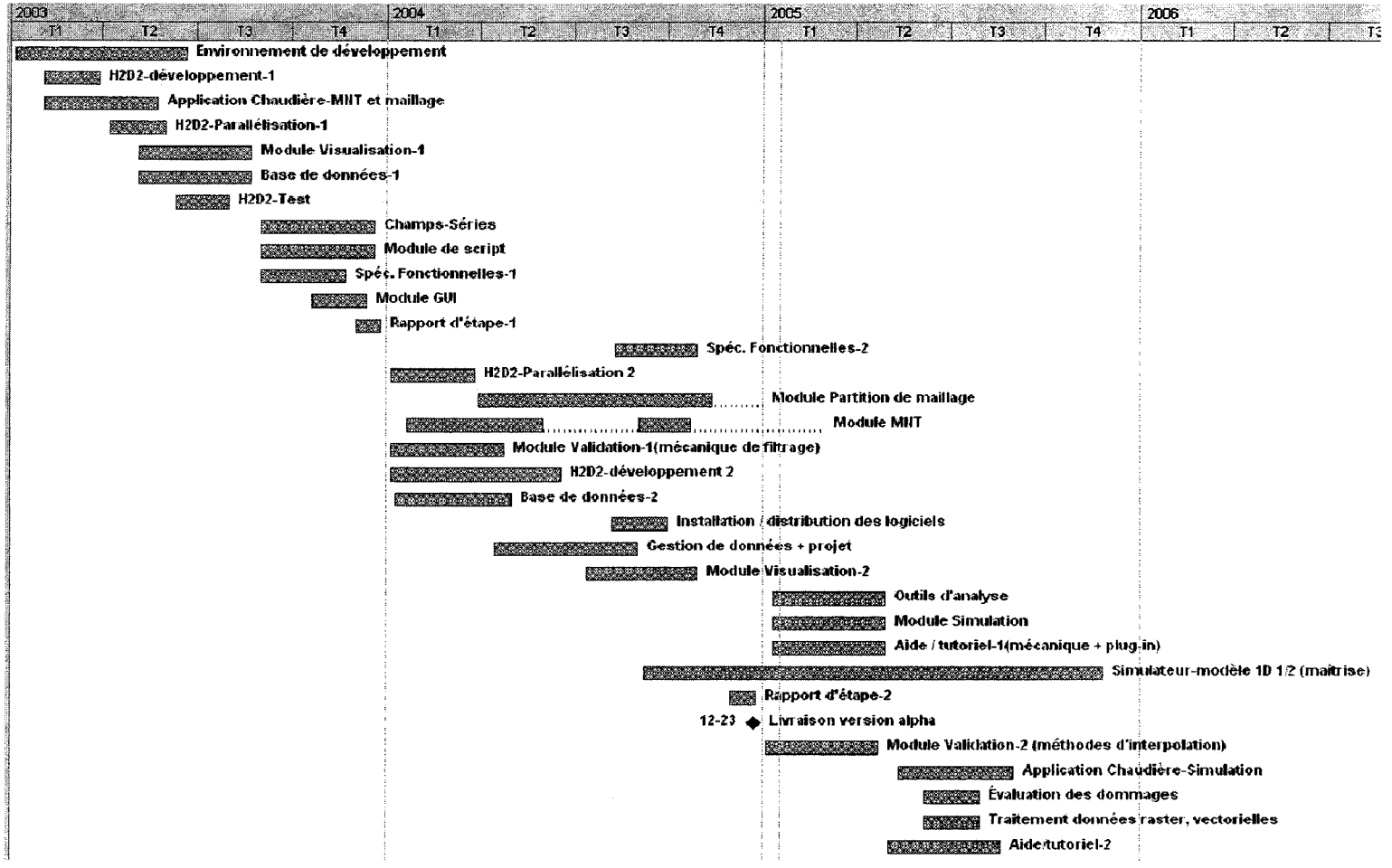
Cette tâche a pour but d'intégrer à Modeleur2 un support pour les données raster et vectorielles.

Cette tâche est complétée à 50%. Le rapport détaillant le travail effectué sur cette tâche est disponible dans le document Rapports de recherche 2004.

Intégration et test finaux

Cette tâche a pour but de s'assurer que toutes les composantes développées dans le cadre du projet s'intègrent bien. Des tests au niveau système seront développés afin de garantir un logiciel fiable.

6.2 Diagramme des tâches



7 Conclusion

Le projet SEGRI offre des défis très stimulants au niveau du développement logiciel. De ce fait, il s'avère qu'il est facile de motiver les étudiants car les stages offerts sont riches et variés. Ces stages constituent une occasion unique pour eux de travailler sur un projet logiciel d'envergure et de se voir confier une gamme importante de responsabilités.

Bien qu'un soin méticuleux soit apporté à l'exploitation optimale des fonds disponibles, la stratégie d'employer uniquement des stagiaires atteint parfois ses limites. On se rend compte que, jusqu'à un certain point, le développement est à la merci des compétences des stagiaires que nous sommes en mesure de recruter. Ceci a pour effet que le développement n'est pas toujours aussi soutenu qu'il pourrait l'être, et qu'il nous est parfois nécessaire de retravailler certaines parties de code au départ d'un stagiaire.

En ce qui a trait à l'aspect planification, le projet avance à un bon rythme. Un léger réordonnancement des tâches a dû être effectué pour tirer profit des compétences diverses des stagiaires. Le calendrier a ainsi été remanié pour prendre en compte ce facteur, mais les impacts sont relativement mineurs sur le plan initial.

Les développements effectués en cours d'année nous donne une plate forme solide sur laquelle construire pour les années à venir. Les éléments clé du logiciel ont été mis en place de façon à ce qu'il soit aisé d'y greffer de nouvelles fonctionnalités et de l'amener à maturité pour une version finale à la fin 2005.

En terminant, il nous semble pertinent de rappeler que sans l'ombre d'un doute, cette méthodologie de développement axée sur des stagiaires a des retombées fort positive pour le Québec de demain. Non seulement, un projet comme le projet SEGRI a pour but d'aider à résoudre des problématiques qui touchent de près la population, mais en plus il contribue à la formation de personnel hautement qualifié.