

Suivi du projet « Ermitage » (5 avril 2008)

Ce suivi qui documente le développement du projet fournit un premier guide d'utilisation du système. Il constitue un mode d'emploi pour la préparation de documents. Il contient un aperçu de la philosophie du projet, une description des interfaces, des modules (logiciels) et des bases de données de même qu'une présentation de différents formats de documents. Puis il présente les différentes étapes à parcourir pour installer pratiquement un document.

La philosophie générale

Le système se base sur une « métaphore » spatiale et propose une approche document.

Ainsi, construire un système c'est donc imaginer une classification qui s'inscrit dans une structure cartésienne (partie x étage) avec pour chaque élément de cet ensemble une liste de zones, chacune d'elles possédant un certain nombre de salles. Il n'est pas impossible que d'autres métaphores s'accommodent de cette structure. Chaque élément est encore caractérisé par un niveau (scolaire). Un utilisateur ne voit que les éléments destinés à son niveau et ceux des environs immédiats.

Introduire une information dans le système, c'est donc situer cette information dans cet espace et écrire le ou les documents contenant l'information. En réalisant un document, on donne également les informations qui permettent de le lier avec d'autres documents.

Les interfaces

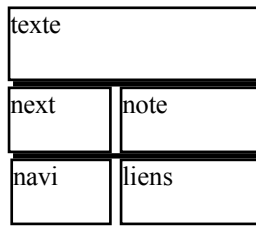
Les interfaces de navigation

Elles dépendent de la structuration des données. Mais en principe, elles sont constituées de quatre zones : déplacement d'une partie l'autre, d'un étage à l'autre, déplacement dans une zone, déplacement dans les salles.

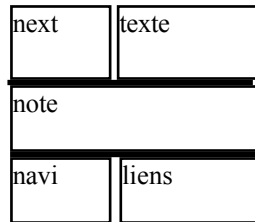
Les interfaces de consultation

Il y a actuellement 11 interfaces possibles pour une salle. Chacune contient 5 zones¹. La zone du texte principal (texte), la zone du texte secondaire (note), la zone des autres documents situés au même endroit (next), la zone de navigation (sortie et déplacement dans une zone adjacente) (navi) et la zone fournissant des liens spécifiques au document (liens). Chacune de ces 5 interfaces a 2 ou 3 paramètres constitués par les dimensions (relatives ou absolues) des fenêtres.

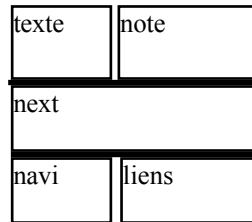
¹ Auxquelles il fait ajouter des zones destinées à afficher des éléments graphiques (actuellement une frise au bas de l'écran).



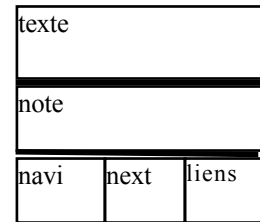
Interface 0 (défaut)



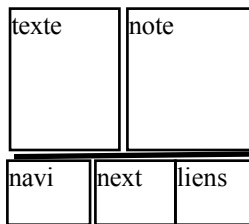
Interface 1



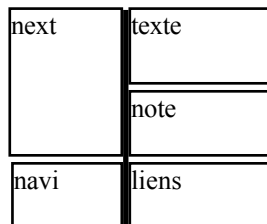
Interface 2



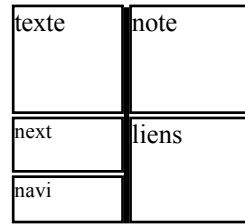
Interface 3



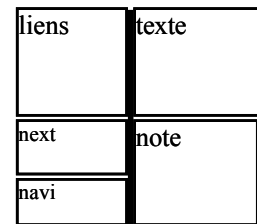
Interface 4



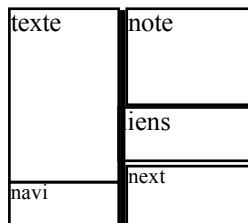
Interface 5



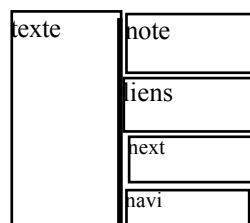
Interface 6



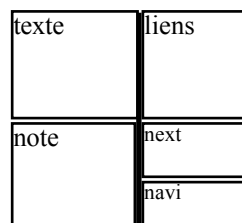
Interface 7



Interface 8



Interface 9



Interface 10

Les modules

Le système est constitué d'un certain nombre de modules, certains sur le serveur, d'autres « embarqués » sur le client.

* Module WXGOTO (module serveur) : ce module gère l'inscription et les déplacements de l'utilisateur dans l' « espace » et génère les écrans de navigation. Il s'occupe également de conserver et d'utiliser le « suivi » de l'utilisateur.

* Module WWWXML (module serveur) : ce module permet à partir des documents écrits en XML de générer des flux HTML à l'intention des navigateurs standard.

* Module des déclarations générales (module client) : il est réalisé en javascript et contenu dans le « frameset » racine. Il contient notamment les déclarations des variables globales de localisation et des tableaux (array) destinés à recevoir des liens.

* Module salle.jx (module client) : ce module est incorporé dans le « frameset » d'une salle. Il contient notamment toutes les fonctions d'affichage. Une partie de ce module est générée, notamment la table des liens qui seront affichés dans les différents « frames » de navigation. Plusieurs des fonctions sont liées à des « éléments » XML dont elles assurent la transcription en HTML.

* Modules spécifiques (modules client) : ces modules sont spécifiques aux documents qui les contiennent. Il s'agit des modules « correcteur » (qcm2.js, qcm3.js), de générateurs de problèmes, de fonctions particulières (pgdc, ...), etc.

* Modules de gestion : deux modules en développement devront faciliter l'entretien de la structure du système. Le premier fournit une interface Web (qui pourra être intégrée aux outils du « conservateur ») qui permet d'insérer un nouveau document au système, l'autre est monoposte. Il concerne des aspects plus techniques (conversion de bases de données). Ces modules faciliteront l'installation d'un nouveau document qui actuellement se fait « à la main ».

L'annexe 1 mentionne quelques modules permettant la migration des données de ProfExpert.

Les bases de données

On décrit ici tous les fichiers et base de données annexes aux modules. Les noms donnés entre parenthèses sont ceux de l'implémentation en cours pour le module de mathématiques.

Le fichier de configuration (webexp.cnf): il contient les chemins des différents composants du système de même que les noms des bases de données.

L'architecture du système (webmath.lex): cette base contient les différents « lieux » du système caractérisés par : leur code (numérique), leur nom abrégé et leur titre.

La liste des documents (webmath.pro) : cette base contient la liste des documents. Un document est caractérisé par : un indice de visibilité (1 : le document est indiqué dans le frame « next », 0 : il n'est pas affiché), le nom du « package » (rmt, mathbas0, etc.), le nom du fichier, le nom du document, son titre, sa position (partie, domaine, sous-domaine, sujet), son type (fiche, exercice, situation, ...), son niveau, et son numéro d'ordre. Un document figurant à deux endroits est représenté par deux enregistrements. Par la suite, il est proposé d'utiliser deux relations (package, fichier, nom, titre) et (nom, visibilité, position, no d'ordre, niveau et type). Ultérieurement, dès que le nombre de documents sera plus élevé (plus de 1000 documents), cette liste sera transférée dans un système permettant l'indexation (webmath.hba).

Les relations (relation.pro) : cette base contient les liens entre les différentes parties (liens entre les « salles », liens entre « salles » et « zones ») Cette base contient également l'interface particulière de chaque salle. Cette base pourra également contenir des conditions particulières pour la visite de certaines zones (pour aller dans la zone X il faut avoir visiter la salle Y et fait l'exercice Z). Les prémisses de ces conditions devront se trouver dans le suivi de l'utilisateur.

A noter qu'une partie des relations sont actuellement incluses dans le module WXGOTO (liens entre zones, domaines, parties).

Les contenus des « panneaux » (panneau.pro) : cette table permet d'indiquer pour chaque endroit (lieu, dom, sdom, level), une liste de fichiers qui fournissent le contenu de panneau. Cette liste est accompagnée d'un type (0 on prend le premier élément de la liste, 1 on prend un élément au hasard, 1 on prend l'élément lié au jour du mois (dans ce cas la liste ne doit pas contenir plus de 30 éléments)). Le panneau est constitué d'un « layer ». D'autres tables de la même base permettent d'introduire une information supplémentaire de style. Elle pourra aussi contenir un élément de scénario. Ces panneaux pourraient être multipliés. A réfléchir à une uniformisation : notamment les contenus des panneaux pourraient être des unités « standard » (c'est-à-dire des flux provenant de fichiers « xml »).

Le module WWWXML possède également deux fichiers de configuration : wwwxml.cnf (chemins) et fuzzy.cfg (éléments HTML qui ne sont pas conformes à la norme XML).

Il y a encore un fichier de mots de passe et un fichier de configuration par utilisateur et un fichier de suivi.

Les documents

L'organisation des documents est contenue dans la DTD PEXP. Ce document est toujours en évolution. L'idée est d'admettre l'usage libre de HTML et de le remplacer petit à petit par des éléments typiques. Par exemple : une définition constituée d'un texte, d'exemples et d'illustrations.

On passera en revue des formats de fiches qui constituent des usages locaux de la DTD. A préciser qu'à chacune de ces fiches correspond un module XSLT dont l'utilité est de transformer le document XML en un « flux » HTML.

Les formats qcm2 et qcm3

Ces formats permettent la création de questionnaire à choix multiples gérés par des fonctions en JavaScript.

Le format qtml

Comme tout document PEXP, un document qtml est divisé en deux parties dont la première est HEAD. La deuxième partie est un élément BODY qui contient du XHTML (qui pourra se spécifier ultérieurement).

L'élément HEAD peut contenir les éléments TITLE, ATTLIST, CTXLIST et LINKS

TITLE : titre du document.

ATTLIST : liste les attributs (caractéristiques) du document. En particulier les attributs from, copyright, reference sont utilisés pour afficher (fonction afficheRef) les références au chargement du document. Ce qui provoque le nettoyage de la zone de notes. Sinon le texte ancien reste affiché (mettre une référence vide si on veut simplement effacer la fenêtre de note, notamment dans les enchaînements).

CTXLIST est la liste des « contextes ». C'est un élément général de PEXP, qui n'est pas activé dans qtml.

LINKS contient la liste des liens « externes » (LNK) ou « internes » (INTLNK).

L'élément LNK (liens externes) est vide. Ces attributs sont :

- TYPE : texte, note, popup, bigPopup
- FUNCTION : exemple, info, demonstration, ..
- HREF : lien
- TARGET : en principe la cible dépend du type et de la fonction, elle peut toutefois être précisée.
- NAME, TITLE : le nom de l'unité d'information et son TITRE.
- IDREF : la référence avec laquelle le lien est mentionné dans le BODY (quelques IDREF sont prédéfinis: exit, onload, error, msg, link).

Si IDREF est vide, le lien figurera dans la fenêtre « liens » (avec TITLE comme hyperchamp). De même si IDREF n'est pas vide et NAME est précédé de @. Au

cas ou HREF n'est pas indiqué, on utilise NAME et on regarde si l'activité est dans la même salle ou non.

Si le nom de la FUNCTION est précédé du signe @ (par exemple @info) on regarde si l'activité est dans la salle ou non. Si tel n'est pas le cas on « changera » de salle lors de l'activation du lien. Autrement dit tous les documents liés sont virtuellement dans la même salle, sauf si:

- 1) réellement ils n'y sont pas et si @ figure au début de la fonction ;
- 2) TARGET se réfère à une autre fenêtre (notamment les liens sur des sites externes).

L'élément INTLNK contient une zone de texte (#PCDATA non balisé) qui est affiché selon les mêmes modalités que l'information liée à LNK.

A noter que l'information des liens devient un tableau JavaScript. LNK devient un objet avec l'attribut place=0, et INTLNK avec place = 1. L'attribut url dans le premier cas contient l'adresse de l'information et l'information dans le deuxième cas.

A prévoir un deuxième type de lien interne (OUTLNK) pouvant contenir des éléments balisés avec, par exemple, la création d'un élément LNK et d'un fichier lors de la compilation.

Remarques :

- NAME est le nom de l'ui, IDREF est le nom du lien !
- Attention lors de l'installation, qtml.xml contient le nom du serveur !

L'élément BODY contient du XHTML (
, etc.). Des appels à des fonctions javascript sont possibles. Ultérieurement le format qtml sera amélioré pour simplifier l'écriture de l'appel de ces fonctions.

Les fonctions utilisables sont les suivantes :

- parent.afficheDirect(ref) : l'information contenue dans l'objet LNK ou INTLNK avec IDREF='ref' est affiché. Actuellement, l'affichage se fait dans la même salle. On adoptera la convention suivante; si NAME n'apparaît pas, on affiche dans la même salle (cas fréquent). Sinon, on utilise la même convention que les liens depuis la fenêtre 'liens' (usage du signe @ devant la fonction).

- parent.affichePopUp(texte) : le texte est affiché dans une petite fenêtre. Attention les pages xml ou html utilisant cette fonction doivent contenir dans leur BODY l'attribut ONFOCUS= « parent.hidepic() ». Si texte contient htm c'est le contenu du fichier htm qui est affiché. Cette fonction permet d'éviter l'usage des éléments LNK. Elle peut donc être utilisée dans de simples pages HTML.

- parent.afficheNote('texte') : texte est affiché dans le frame 'note'

Les deux dernières fonctions peuvent aussi être utilisées aussi bien dans des pages HTML que XML. Les images prennent pour racine dans les pages XML et HTML <serveur>/webexp/base

Note de développement concernant les liens:

- Actuellement écrit directement à l'aide de <A HREF="parent.afficheDirect(<ref>)">; deviendra <A IDREF="<ref>">

- Pour rendre les multi-liens, il y a deux solutions: <ref> peut être une liste de références (cf Hytime). Ou alors il peut y avoir plusieurs éléments <LNK IDREF="<ref>"

qtml-1 contient un élément SCRIPT implémentant afficheCartouche.

Le format vtml

C'est un format identique au précédent avec en plus des variables « utilisateur »² : globales ou locales à une salle.

Les variables apparaissent dans la section VARLIST de l'élément HEAD sous la forme de l'élément VAR (attributs NAME et VALUE) pour les variables globales et LOC (avec les mêmes attributs) pour une variable locale.

VALUE est évalué avant l'attribution. Toute « chaîne de caractère » (string) est considérée comme une fonction.

Outre les fonctions de JavaScript, les fonctions suivantes sont définies qui peuvent être utilisées:

- vL(nvar) : renvoie la valeur de la variable (LOC) nvar
- valLoc(nvar) : identique à la précédente (si non définie renvoie son nom)
- valVar(nvar) : idem pour une variable globale (si non définie renvoie "")
- int_rnd(int min, int max): renvoie une valeur comprise en min et max (y compris)
- char_rnd(str str) : renvoie un caractère de la chaîne str.
- tab_rnd(Array tab) : renvoie un des éléments du tableau tab.
- attrib(int n,str, array [var1,var2, ...]) : la fonction renvoie true avec un effet de bord qui attribue la valeur str à n variables de la liste. str est évalué à moins que la chaîne ne débute par \$ (qui n'est pas pris).

D'autres fonctions sont définies qui interviennent de façon plus technique lors de la programmation de nouvelles feuilles de styles (voir annexe 8).

Pour afficher la valeur d'une variable dans une page on pourra utiliser l'appel : parent.vL(var) contenu dans une balise SCRIPT. Comme précédemment, une version ultérieure permettra de simplifier l'écriture du BODY (voir mathba1b/ui_re2x2.xml et mathba1b/ui_ad2x2.htm).

L'élément HEAD peut également contenir un élément SCRIPT qui sera recopié dans le « flux » HTML. (ultérieurement cette partie pourrait être reformulée dans la DTD PEXP).

Les annexes 2, 3 et 4 présentent respectivement : le script vtml.xsl, un exemple de fichier XML faisant usage de ce script et un document HTML utilisant les variables initialisées dans le document XML.

vtml-1 implémente la fonction printLoc.

vtml-1a met la feuille de style de note à la place de la feuille de style standard.

Le format fiche

Le format fiche contient une partie d'entête comme précédemment. Sa deuxième partie est l'élément CONTENT constitué d'un élément INTRO puis d'une suite d'éléments DEFINITION.

² Ne pas confondre avec les variables systèmes : globales : wxXXX, locales à une page : lcXXX

Chaque élément DEFINITION contient une partie TEXT, une liste d'éléments optionnels EXAMPLE et d'un élément (optionnel) BBOARD.

BBOARD est un élément graphique pouvant être configuré. Ses attributs sont HEIGHT, WIDTH et PRGM. PRGM contient une chaîne représentant la fabrication d'un dessin dans un langage s'apparentant à logo (cf annexe 5). Le contenu de BBOARD consiste en un élément CAPTION et d'une suite d'éléments LABEL (voir mathbas0/ui_196.xml).

Le format fiche2

Ce format étend le format fiche (en cours de développement) (voir mathbas1/er11-001.xml).

Le format lacune

Ce format qui pourra aussi être intégré à 'fiche' permet d'insérer des APPLLET d'interaction LACUNE. L'annexe 6 ébauche une structure générale pour les applets d'interaction.

Le format qrep

Ce format est prévu pour créer des chaînes de questions (voir RALLYE).

Il doit avoir la déclaration de variables locales : no_rep à 0 (numéro de l'essai courant), max un nombre ou « infini » (nombre d'essais possibles) et un modèle (donne la manière de calculer la réponse dans certains cas).

Une partie SCRIPT contient la fonction verifReponse(sol) et les fonctions auxiliaires.

Les liens (IDREF) sont « link » (question suivante), « error » (message d'erreur), et « msg » (message complémentaire).

- "link" est déclenché lorsque la réponse est juste
- "error" lorsqu'il y a une erreur
- "msg" est un message complémentaire (par exemple combien il reste d'essais). Cela peut aussi cumuler error et complément (idem pour error).
- "exit" est déclenchée lorsque le nombre d'essais dépasse le nombre maximum. Si ce lien n'existe pas la fonction "sortie()" est déclenchée à la place.

A noter que ces déclencheurs peuvent être utilisés au-delà de l'environnement qrep, la plupart étant utilisés à partir de fonction de salle.jx.

Un autre idref général est 'onload'. Le lien est enclenché au chargement de la page (à partir de la fonction afficheRef()). (voir problem1/ui_286-*.xml pour un exemple d'utilisation).

La variable javascript locReponse contient la réponse (utilisable dans le feedback).

L'élément CONTENT contient :

- du code HTML dans les éléments DIV
- un élément INTRO qui peuvent contenir des éléments VAL-VAR (valeur de variables locales, fonctions, images, etc., cf pexp3.txt)

- un élément FORM-REP pour le formulaire de la réponse. Les attributs de ce formulaire sont SIZE (largeur de la zone de réponse), TYPE et SCHEME. Son contenu est un texte pouvant contenir des éléments VAL-VAR.

TYPE	SCHEME	Signification
Entier / decimal	apply	La réponse doit être un entier / decimal, la vérification fait appel à la fonction verifReponse
Entier /decimal	textuel	La réponse doit être un entier /decimal, la vérification se fait directement, sans évaluation selon le modèle
bouton	apply	verifReponse est appliquée sans valeur.
		Sinon verifReponse est appliquée à la réponse brute.

Une fonction simpleReponse(type, scheme, reponse) peut être utilisée directement, par exemple sur une image munie d'une ancre A.

Le contrôle se fait via controleReponse(<formulaire>).

Développements futurs :

- Ajouter d'autres TYPE/SCHEME (par exemple "texte")
- Types et schemes permettant de montrer une partie d'image
- Variables de cumul (suivi) (fait en partie 03.03)
- Examen du problème de la touche return. Pour le moment déconnectée grâce à un ONSUBMIT="return false". (voir la solution dans les flash)
- Pour l'envoi d'un formulaire, il faut pour le moment le créer soi-même dans un document xml avec une ACTION contenant explicitement un script et un champ caché de nom 'idref' et de valeur la référence au document (htm ou xml) à envoyer comme quittance. Le document doit aussi contenir un élément LNK avec l'IDREF correspondant.
- La fonction du script est de traiter (d'enregistrer) les données reçues et d'envoyer une page contenant seulement l'appel à la fonction parent.afficheDirect(<valeur de idref>). Cela marche parce que le tableau des liens reste en mémoire tant qu'une nouvelle page xml (ou une page contenant une remise à zéro de la base) n'est pas appelée.
- Élément pour un formulaire d'envoi, voir qrep-fls pour des idées

Extension qrep2 et qrep2-1

Version qrep2-2

Cette nouvelle version prend en compte l'attribut TYPE de l'élément VAL-VAR qui affiche la valeur de la fonction définie librement dans la zone script.

De plus VAL-VAR prend un attribut FORMAT qui permet d'encapsuler des scripts générant du code HTML (par exemple : document.write(makeImageEI) correspondant à <VAL-VAR FORMAT= « img »/> ; cf itinerai/md_pb3.xml, voir aussi le modèle dftml).

Cette zone script a été placée après les scripts de définition des variables. peut prendre l'attribut STYLE. (version 08.02.03).

Ajout des éléments BR, BLOC / LIGNE | CASE et DIV à l'intérieur d'INTRO.

Les contextes sont mis en oeuvre de façon restreinte (cf `itinerai/md_df1.xml`). Il faudrait ajouter les automatismes de mise à jour dans `vtml-2`. De même pour les initialisation (`md_df0.xml`). Ils utilisent les variables « utilisateurs » (LOC max, no_rep, poids, penalite) ce qui devrait être remplacé par des variables systèmes `lc*`. Ce point est à étudier en même temps que le problème global de la mise à jour des variables no_rep, poids, cumul, cumul_max, suivi, no_qs (assurée actuellement par les pages de messages, cf `itinerai/md_df0.xml`, `itinerai/md_df1.xml`, `itinerai/md_adf1.xml`, `itinerai/df_msg.htm`, `itinerai/df_imsg.htm`). Usage également de `poids_init` pour conserver le poids initial de la question.

Au démarrage, `init_page` appelle `afficheLink()`, `afficheRef2(.,.,.)` (sans `onload`), `parent.afficheIntro()` et `init_focus()`. `afficheRef2` (contrairement à `afficheRef()`) n'efface pas la fenêtre des notes s'il n'y a pas de référence (donc mettre un `copyright = « »` pour effacer la fenêtre).

Les fonctions `makeInput(type,name,value)`, `makeInput1(type,name,size)`, `makeTR(nbr_cell,tab_cell)` pourront être ultérieurement encapsulées dans de nouveaux éléments XML.

La même version a été enrichie d'une possibilité de STYLE.

Associés fortement à ce modèle les documents de feed-back `$xml/util/smsg-ok.xml` et `$xml/util/smsg-nok.xml` génèrent des messages plus ou moins variés. `$doc/util/suivi-ok.htm` et `$doc/util/suivi-nok.htm` améliorent les documents de feed-back `smsg-ok` et `smsg-nok` (ajoute le formulaire de suivi dans le cas de ok et permet des messages différenciés pour nok). (voir `$xml/calcul3/cpi-exe-cal01.xml`). Le passage des fichiers de `$doc` à la place de `$xml` permet de garder l'identifiant de l'ui principale en vue de la création de l'enregistrement du suivi. Attention à gérer la variable locale « msg » (si différente de « » est affichée).

Le format vtml-1d

Format vtml auquel sont ajoutées l'initialisation des variables nécessaires aux « piste matematica » (`cumul`, `cumul_max`, `suivi`, `no_qs`) (cf annexe 11)

Le format vtml-2, vtml-2p

Extension de vtml. Introduit les objets NOTE et CARTOUCHE qui sont à préférer à A. BLOC / LIGNE | CASE. (`<H IDREF=>` est aussi là en partie double emploi avec NOTE). Est le pendant de `qrep2-2`.

Au démarrage, même `init_page()` que dans `qrep2-2`. *vtml-2p* cumule les liens (prévu pour des relances de problème).

Le format vtml-2s

Reprise du précédent pour les documents « principaux » (style standard). Mis à la racine xml.

Tableau résumé des formats de documents

Modèle	Contenu	Style	Liens	Remarque
Qtml	body	standard	remplace	
qtml-1	body	standard	remplace	avec cartouche

Vtml	body	standard	remplace	obsolète
vtml-1	body	standard	remplace	
vtml-1a	body	Note	remplace	
vtml-1p	body	Note	cumule	
vtml-1d	body	standard	remplace	initialisation défi
vtml-2	content	Note	remplace	
vtml-2p	content	Note	cumule	
vtml-2s	content	standard	remplace	

Attention : dès 2004, les modèles sont à la racine xml. A version égale, il peut y avoir des modifications légères avec les fichiers aux anciens emplacements (place de la copie des scripts notamment).

Le format qrep-exp

Identique à qrep2-1 mais charge (ev. à ne faire que le lien) les fonctions Javascript selon le contexte d'expertise.

Voir `problem1/ui_1898b.xml` pour un premier exemple qui fait appel à `proport.js` pour le contexte d'expertise concernant la proportionnalité. `problem1/ui_1898.xml` est plus complexe.

Ce format sera encore étendu pour mettre en oeuvre d'autres contextes.

Le format qrep-qcm

Spécialisé pour qcm, fait la synthèse de qcm2/3, qrep-exp, qrep2-2. Meilleur usage des contextes. Ajoute la variable `nb_essais` qui garde le nombre d'essais même si `no_rep` est remis à 0.

Le format qrep-fls

Spécialisé pour flash. Utilise `flash.js`. Introduit la possibilité d'enregistrer (suivi) le score obtenu à la fin de l'exercice. La touche « return » permet de valider la réponse (voir `qrep-fls.xml`). Ces idées sont à implémenter dans les autres modules de qrep (suivi fait pour `proport.js`, juin 2003). La possibilité d'utiliser l'interaction flash dans les défis a été introduite (janvier 2007). Pour cela le contexte contrôle/mode doit être mis à défi. Le contrôle `nbReprise` donne le nombre de fois où le « paquet » peut être repris, lorsque après l'avoir passé en revue, le nombre d'erreur est trop grand.

Le format dftml

Ce format est conçu pour la mise en page des feed-back. Il prend en charge les variables prédéfinies `<VAL-VAR NAME=#locReponse/>`³ et `#bilan`. Il charge également le javascript nécessaire au suivi. (voir annexe 11)

Résumé de la galaxie qrep

Modèle	Contenu	Style	Liens	Remarque
qcm2				obsolète
qcm3				obsolète
Qrep	content	standard	Remplace	obsolète

³ signifie `<SCRIPT LANGUAGE="JavaScript">document.write(parent.locReponse);</SCRIPT>` où `locReponse` est une variable JavaScript définie dans `salle.jx` (une couche au-dessus). `#bilan` fabrique un bilan de manière adhoc avec les éléments locaux.

qrep2	content	standard	Remplace	
qrep2-1	content	standard	Remplace	
qrep2-2	content	standard	Remplace	
qrep-qcm	content	standard	Remplace	
qrep-fls	content	standard	Remplace	
qrep-exp	content	standard	Remplace	
Dftml	content	standard	Remplace	

Résumé des différentes fiches

Modèle	Contenu	Style	Liens	Remarque
Lacune		standard	remplace	
Fiche		standard	remplace	
fiche2		standard	remplace	
Applet		standard	remplace	
Applet2		standard	remplace	

Le format activite

Ce format utilise des layers pour placer les différents zones : consigne, énoncé, message, activité. Les autres formats pourraient s'en inspirer notamment qrep-exp pour améliorer la disposition. A améliorer et compléter lorsque d'autres activités que « course de nombres » (jeux/course.xml) auront été mises au point.

Le format qtml-2

Améliore le format qtml et facilite l'écriture des fiches théoriques. Le corps est constitué d'un élément « CONTENT ». Un minimum d'éléments HTML sont possibles avec en plus les éléments NOTE et CARTOUCHE (repris de vtml-2).

Mises à jour

Les mises à jour se font uniquement avec les feuilles xml à la racine xml. Attention : il en découle qu'il peut cohabiter des feuilles de style différentes portant le même nom.

qrep-exp : possibilité de mettre un style dans la consigne générale (juin 04).

Le format touchpad

Utilise les formats « activites » et « qrep-fls » pour faire des activités où il s'agit de montrer des zones d'images. Introduit le flash « infini » et le passage des items définis comme des <LOC> à des éléments du contenu <ITEM>, ce qui demande une double utilisation des éléments (alimentation d'objets jvs et mise en place d'éléments HTML). (voir qcm0/ui_cc01.xml). A faire évoluer pour prévoir des variantes commandées par le contexte (CTXT).

Le format qcm-img3

Inspiré du précédent (devrait être différencié par CTXT), il permet de cliquer sur un élément (image ou zone de texte) pour désigner une réponse à un énoncé. (voir qcm0/ui_cc02.xml)

Le format qrep-mult

Fabriqué à partir de qrep-qcm, ce format permet de remplir plusieurs zones de texte. A voir pour lier un suivi (à faire de même pour qrep-qcm). Voir aussi pour relier ce format à d'autres via les CTXT. (voir qcm3/ui_prob1.xml)

Le format puzzle

Ce format regroupe différentes interactions (via CTXT) liées à des « cartes » alignées en tableau. Trois types existent : memory/1 (memory classique), memory/2 (les deux cartes de la paire peuvent être différentes), puzzle (images à échanger).

Dans chaque cas les informations à donner dans le contexte sont le nombre de lignes et de colonnes, le format des images (largeur, hauteur), le nom générique (kName) des images et le répertoire (kPath).

Les images sont à mettre dans \$html/\$package/images/\$kPath. Dans le cas de memory \$kName{0}.gif est le dos des cartes. Les images sont \$kName{i}.jpg (et \$kName{i}a.jpg lorsque la 2^e carte de la paire est différente dans le cas de memory/2). La fabrication d'un memory/3, etc. ne demande que des retouches mineures au script puzzle.js.

Dans le cas de puzzle/1, les images commencent par \$kName{0}.jpg. Pour fabriquer les pièces du puzzle, on pourra s'aider d'un « splitter » d'images (par exemple : ImageCut).

Le format touchpad2

Ce format reprend touchpad en utilisant le cumul. C'est-à-dire que le clic sur une image ajoute la valeur associée à la valeur précédente (voir \$xml/cpi/exemonp10.xml). Les deux modèles utilisent touchpad.js, le premier avec interaction/mode map/simple et l'autre map/cumul. Ce format ajoute les attributs permettant de fixer la grandeur des zones d'interaction.

Les algorithmes

Création d'une interface de navigation

Cette création est provoquée par une requête sur WXGOTO avec les paramètres : dep=partie, les coordonnées du lieu où l'on se rend et le profil de l'utilisateur (sobriquet, langue, niveau, « largeur »). Le module va puiser dans l'architecture (lex) pour afficher partie et zones.

Création d'une salle

Elle est provoquée par une requête sur WXGOTO avec les paramètres : dep=salle. La situation est un peu plus complexe. Trois cas sont considérés :

1. Les coordonnées de la salle sont données avec le nom de l'activité
2. Les coordonnées de la salle sont données sans le nom de l'activité : dans ce cas on sélectionne l'activité de la salle commençant par « index » ou sinon la première activité présente et on se ramène au cas 1.
3. Seul le nom de l'activité est donné : dans ce cas on recherche la salle (actuellement la première où se situe l'activité) et on se ramène au cas 1. Ce choix de la salle (cas assez rare) reste un problème ouvert.

Ce module va générer des tableaux JavaScript qui contiennent la liste des salles liées (relation.pro) (table tabAdj), l'existence d'une sortie éventuelle, la table de

toutes les activités de la même salle qu'il faudra afficher (table tabActAdj) et une table contenant l'ensemble des activités de la salle (y compris celles que l'on ne peut atteindre que par des liens à partir d'autres activités) (table tabAct). Puis il va envoyer le « frameset » correspondant à cette salle.

Le frame principal ('texte'), contient comme « source » (SRC) une requête sur le fichier html ou xml du document d'introduction. Les autres sont vides et vont être initialisés par des scripts.

Le frame secondaire ('note'), est en principe utilisé comme cible des liens liés à l'information principale. Au chargement, il contient les références du contenu principal (à introduire la possibilité de faire charger directement un lien désigné par défaut).

Le frame 'next' enclenche la fonction afficheRoomCtxt() qui elle-même lance afficheActAdj (fabrication du contenu du frame 'next') et afficheAdj (fabrication du frame 'navi').

Le frame 'liens' contient les liens liés au document principal.

L'affichage de la fenêtre 'lien' et éventuellement 'note' est toujours lié au chargement de l'information dans la fenêtre principale « texte » (cf. Chargement d'une activité dans une salle).

Les liens dans le frame 'navi' sont toujours de type : dep=salle. Les liens de 'next' sont des liens directs sur des documents xml ou html.

Le frame 'frise', enclenche affFrise(), qui elle-même appelle afficheFrise avec des paramètres (frise, couleur, etc.)

Chargement d'une activité dans une salle

Il s'agit du chargement du contenu de la zone principale 'texte' qui peut se faire à l'entrée dans la salle ou lors d'actions sur des liens de la zone 'next' (éventuellement de la zone 'lien' lorsque le type est 'texte').

Si la page est une page HTML en principe rien n'est modifié hormis la fenêtre 'texte' (mais la page HTML peut contenir des scripts ad hoc).

Si la page est un document XML, un certain nombre de procédures standard existent dont l'appel est à insérer dans la feuille XSL.

1. L'affichage d'informations de référence dans la zone 'note' (la fonction afficheRef utilise les attributs from, copyright et reference du document). Cette fonction lance également, le cas échéant, le lien 'onload'.
2. L'affichage de la liste des liens dans la zone 'liens'.

On voit donc qu'à la racine, la seule fonction existante est le chargement de documents. Ensuite chacun d'eux se débrouille via sa feuille de style ou en faisant appel à des fonctions javascript globales (ermitage.htm), semi-globales (salle.jx) ou locales (.js associé à la feuille de style).

Installer un nouveau document

Pour installer un nouveau document les étapes à parcourir sont les suivantes (en attendant un système automatisant la procédure) :

1. Rédiger la fiche XML ou HTML

2. La placer dans un répertoire (package)
3. Faire une nouvelle entrée dans le fichier webmath.pro
4. Le cas échéant (par exemple création d'une nouvelle salle ou zone) créer les rubriques nécessaires dans le fichier webmath.lex
5. Compléter en cas de création d'une nouvelle salle le fichier relation.pro

Le nom de la fiche de présentation de la salle doit commencer par 'index'.

Références

Pochon, L.O. & von Siebenthal, C. (1994). *Experts, contexte et événements : description du système Prof'Expert pour le programmeur*. ABORD.

Pochon, L.O. (2000). *Le projet « Ermitage » : définition d'un environnement hypertextuel d'apprentissage*. Neuchâtel : IRDP.

Annexe 1 : Migration depuis Prof'Expert

Les étapes permettant de récupérer les données de Prof'Expert sont les suivantes :

1. Le module 'mouline' demande un nom de base <base> et convertit <base>.hba en deux fichiers : <base>.dat qui contient les nodes et <base>.txt qui contient les textes.
2. Avec un éditeur DOS ; il faut remplacer les caractères de saut conditionnel (note !) par un espace.
3. Le module toxml.pro permet de générer les fichiers XML (un par unité d'information) qui vont se loger dans le répertoire <base>.

Annexe 2 : Le script vtml.xsl

```
<?XML VERSION="1.0"?>
<!DOCTYPE xsl:stylesheet SYSTEM "xsl.dtd">

<xsl:stylesheet>
<xsl:output doctype-public="-//W3C//DTD HTML 4.0 Transitional //EN"/>

<xsl:template match="/">
<HTML>
  <xsl:apply-templates/>
</HTML>
</xsl:template>

<xsl:template match="PEXP">
<HEAD>
  <xsl:apply-templates mode="head"/>
  <BASE HREF="http://<serveur>/irdp/webexp/{$base}/info.html"/>
  <LINK HREF="../styles/standard.css" REL="stylesheet" TYPE="text/css"/>
</HEAD>
  <xsl:copy-of select="BODY"/>
</xsl:template>

<xsl:template match="HEAD" mode="head">
<xsl:apply-templates/><!-- ATTLIST -->
<TITLE><xsl:value-of select="$name"/></TITLE>

<SCRIPT LANGUAGE="JavaScript">

var ind_link = 0 ;
<!-- var ind_int_link = 0; -->

<xsl:apply-templates mode="links"/>

parent.parent.wxPtTabLink = ind_link ;
<!-- liens internes
parent.parent.wxPtTabIntLink = ind_int_link ;
-->

<!-- autres activites (next) salles ou ailleurs -->
parent.afficheLink() ;

parent.parent.wxCurrentUi = "<xsl:value-of select="$name"/>";

parent.afficheRef('<xsl:value-of select="$ref"/>', '<xsl:value-of
select="$cop"/>', '<xsl:value-of select="$from"/>') ;

<!-- variables globales et locales -->

<xsl:apply-templates mode="varlist"/>

function vV(nvar) {
  return parent.parent.valVar(nvar) ;
}

function vL(nvar) {
  return parent.valLoc(nvar) ;
}

</SCRIPT>
</xsl:template>

<xsl:template match="ATTLIST">
<xsl:for-each select="ATT">
  <xsl:if test="@NAME='name'">
    <xsl:variable name="name" select="@VALUE"/>
    <!-- xsl:value-of select="@VALUE"/ -->
  </xsl:if>
  <xsl:if test="@NAME='base'">
    <xsl:variable name="base" select="@VALUE"/>
  </xsl:if>
  <xsl:if test="@NAME='copyright'">
```



```

        <xsl:variable name="cop" select="@VALUE"/>
    </xsl:if>
    <xsl:if test="@NAME='reference'">
        <xsl:variable name="ref" select="@VALUE"/>
    </xsl:if>
    <xsl:if test="@NAME='from'">
        <xsl:variable name="from" select="@VALUE"/>
    </xsl:if>
</xsl:for-each>
</xsl:template>

<xsl:template match="LINKS" mode="links">
<xsl:for-each select="LNK">
parent.parent.wxTabLink[ind_link++] = new parent.parent.wxLinkObj("<xsl:value-of
select="@TYPE"/>","<xsl:value-of
select="@IDREF"/>","<xsl:value-of select="@NAME"/>","<xsl:value-of
select="@TITLE"/>","<xsl:value-of select="@HREF"/>","<xsl:value-of
select="@TARGET"/>","<xsl:value-of
select="@FUNCTION"/>","<xsl:value-of select="@LEVEL"/>",0);
</xsl:for-each>

<xsl:for-each select="INTLNK">
parent.parent.wxTabLink[ind_link++] = new parent.parent.wxLinkObj("<xsl:value-of
select="@TYPE"/>","<xsl:value-of
select="@IDREF"/>","<xsl:value-of
select="@NAME"/>","<xsl:value-of
select="@TITLE"/>","<xsl:value-of strip-space="yes" select="."/>","<xsl:value-of
select="@TARGET"/>","<xsl:value-of
select="@FUNCTION"/>","<xsl:value-of select="@LEVEL"/>",1);
</xsl:for-each>

</xsl:template>

<xsl:template match="VARLIST" mode="varlist">

<xsl:for-each select="VAR">
parent.parent.newVar("<xsl:value-of select="@NAME"/>","<xsl:value-of
select="@VALUE"/>") ;
</xsl:for-each>

<xsl:for-each select="LOC">
parent.newLoc("<xsl:value-of select="@NAME"/>","<xsl:value-of
select="@VALUE"/>") ;
</xsl:for-each>

</xsl:template>

</xsl:stylesheet>

```

Annexe 3 : Un fichier XML utilisant le script vtml.xml

```
<?XML VERSION="1.0"?>
<!DOCTYPE PEXP SYSTEM "pexp.dtd">
<?XML-STYLESHEET TYPE="text/xsl" HREF="vtml.xml"?>
<?XSL-FLAG VALUE="html-output"?>
<PEXP>

<HEAD>
<TITLE>R&eacute;solution d'&eacute;quations du 2<SUP>e</SUP> degr&eacute;</TITLE>
<ATTLIST>
<ATT NAME="base" VALUE="mathbalb"/>
<ATT NAME="file" VALUE="ui_1deg2.xml"/>
<ATT NAME="level" VALUE="6"/>
<ATT NAME="name" VALUE="ui_1deg2"/>
<ATT NAME="dtype" VALUE="exercice"/>
<ATT NAME="copyright" VALUE="(c) LOP, 1999"/>
<ATT NAME="keywords" VALUE=""/>
</ATTLIST>

<VARLIST>
<LOC NAME="a" VALUE="int_rnd(1,10)"/>
<LOC NAME="b" VALUE="int_rnd(1,12)/2"/>
<LOC NAME="c" VALUE="int_rnd(1,20)/2"/>
<LOC NAME="sa" VALUE="char_rnd('+-')"/>
<LOC NAME="sb" VALUE="char_rnd('+-')"/>
<LOC NAME="sc" VALUE="char_rnd('+-')"/>
</VARLIST>

<LINKS>
<LNK NAME="ui_1deg2" HREF="mathbalb/ui_1deg2.xml" TITLE="Autre &eacute;quation &agrave;
r&eacute;soudre" FUNCTION="exercice" TYPE="texte"/>
<LNK NAME="ui_1deg2_sol" HREF="mathbalb/ui_1deg2.htm" TITLE="Solution en utilisant la
formule" FUNCTION="fiche" TYPE="note"/>
</LINKS>
</HEAD>

<BODY STYLE="background: white; font-family: arial, helvetica, sans-serif;">
<H3>R&eacute;solution d'&eacute;quations du 2<SUP>e</SUP> degr&eacute;</H3>

<P>A r&eacute;soudre:</P>
<P ALIGN="center">
<SCRIPT LANGUAGE="JavaScript">
  document.write(" " + vL("sa") + vL("a")) ;
</SCRIPT>
x<SUP>2</SUP>
<SCRIPT LANGUAGE="JavaScript">
  document.write(" " + vL("sb") + vL("b") + "x " + vL("sc") + vL("c") + " = 0 ") ;
</SCRIPT>
</P>

</BODY>
</PEXP>
```

Annexe 4 : Une fiche HTML réutilisant les variables initialisées par la fiche précédente

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML>
<HEAD>
<TITLE>ui_1deg2_sol</TITLE>
<META NAME="dtype" CONTENT="corrige">
<LINK HREF="../styles/corrige.css" TYPE="text/css" REL="stylesheet">

<SCRIPT LANGUAGE="JavaScript">

var methode = "" ;
var resultat1 = "" ;
var resultat2 = "" ;
var a,b,c ;
var sa, sb, sc ;
var disc ;
var rd , x1, x2 ;

function init() {
  a = vL("a") ; b = vL("b") ; c = vL("c") ;
  sa = vL("sa") ; sb = vL("sb") ; sc = vL("sc") ;

  if(sa=='-') { a = 0-a ; }
  if(sb=='-') { b = 0-b ; }
  if(sc=='-') { c = 0-c ; }

  disc = b*b - 4*a*c ;
}

function solution() {
  if (disc<0) {
    methode = "Pas de solution; discriminant = " + disc + " < 0" ;
  } else if (disc==0) {
    x1 = (0-b)/(2*a) ;
    methode = "1 solution; discriminant = 0" ;
    resultat1 = "x<SUB>1</SUB> = x<SUB>2</SUB> = " + fraction(0-b,2*a) + " = " + x1 ;
  } else {
    rd = Math.sqrt(disc) ;
    x1 = (0-b + rd)/(2*a) ;
    x2 = (0-b - rd)/(2*a) ;
    methode = "2 solutions; discriminant = " + disc + " > 0";
    resultat1 = "x<SUB>1</SUB> = (" + (0-b) + " + <IMG SRC='../images/symboles/racine.gif'
width='18' height='21'>"+ disc + ")/(" + 2*a + ") = " + x1 ;
    resultat2 = "x<SUB>2</SUB> = (" + (0-b) + " - <IMG SRC='../images/symboles/racine.gif'
width='18' height='21'>"+ disc + ")/(" + 2*a + ") = " + x2 ;
  }
}

function vL(nvar) {
  return parent.valLoc(nvar) ;
}

function enforme(n,d) {
  if(d==1) {
    return n ;
  } else if (d==-1) {
    return 0-n ;
  } else if (n==0) {
    return 0 ;
  } else {
    var n1 = Math.abs(n) ;
    var d1 = Math.abs(d) ;
    if (n*d>0) {
      return "" + n1 + "/" + d1 ;
    } else {
      return "-" + n1 + "/" + d1 ;
    }
  }
}
}
```

```

function fraction(n,d) {
    var pg = pgcd(Math.abs(n),Math.abs(d)) ;
    return enforme(n/pg,d/pg) ;
}

function pgcd(n,d) {
    if(n % d == 0) {
        return d ;
    } else if (d % n == 0) {
        return n ;
    } else if (n>d) {
        return pgcd(d,n-d) ;
    } else {
        return pgcd(n,d-n) ;
    }
}

init() ;
solution() ;

function afficheIf(intro,str) {
    if(str != "") {
        document.write(intro+str) ;
    }
}

function writeSol() {
    document.write("<UL>");
    afficheIf("<LI>Conclusion: ",methode);
    afficheIf("<LI>Solution 1: ",resultat1);
    afficheIf("<LI>Solution 2: ",resultat2);
    document.write("</UL>");
}

</SCRIPT>
</HEAD>

<BODY>
<H3>R&eacute;solution d'&eacute;quations du 2<SUP>e</SUP> degr&eacute;</H3>

<SCRIPT LANGUAGE="JavaScript">
    writeSol() ;
</SCRIPT>

</BODY>
</HTML>

```

Annexe 5 : L'objet BBOARD

```
<BBOARD WIDTH="200" HEIGHT="100" PRGM="clear pu re 55 pd sc green av 110 fl
10 ga 270 pu av 5 ga 270 pd sc red av 40 fl 10 pu av 70 ga 90 av 15 ga 90
pd sc green av 70 fl 10">
<CAPTION>(+11) + (-4) = 11-4 = 7</CAPTION>
<LABEL POS="-40,10,green">+11</LABEL>
<LABEL POS="30,10,red">-4</LABEL>
<LABEL POS="-10,-40,green">+7</LABEL>
</BBOARD>
```

Cet élément est à étendre.

Annexe 6 : **Les interactions**

Cette annexe analyse les interactions de Prof'Expert (methode HOOD) en vue de la création de modules java

Dispositif: lors d'une requete sur une page xml de type exercice un programme cgi envoyé une page html avec une applet liee a l'interaction proposée (soit une applet par interaction soit une applet generique avec un paramètre contenant (en plus du fichier des données) les caractéristiques de l'interaction).

Le programme cgi crée également un fichier contenant les données (mises dans une forme pratique pour le transfert et une analyse en java). Si ce fichier existe déjà avec une date plus récente que le fichier d'origine cette création n'a évidemment pas lieu.

Module (interaction)

vi: type ; mode (si applet generique)

meth:

Donnees

vi: raw_data ;

meth: mettre_en_forme ; afficher

Question

vi:

meth: evaluer ,

Reponse

vi: raw_answer ;

meth: analyser ;

Evaluation

vi :

meth :

Question, Reponse ... pourraient etre des interfaces implementees de facons diverses selon le type d'interaction. Idem pour Donnes (mais depend egalement de la donnee (expr math, texte, ...)).

Voir : Pochon, L.O. & von Siebenthal, C. (1994). *Experts, contexte et événements : description du système Prof'Expert pour le programmeur*. ABORD.

Annexe 7 : Installation

IIS / NT

1) Considérer les répertoires (avec des exemples)

\$SYS quelconque ;

\$ROOTSYS: d:\inetpub\scripts ; \$ROOTDOC: d:\inetpub\wwwroot

\$PATHAPP: irdp\webexp ; \$PATHDOC: irdp\webexp

\$APPLI: \$ROOTSYS\ \$PATHAPP ; \$XML: \$ROOTSYS\ \$PATHAPP\xml\

\$DOC: \$ROOTDOC\ \$PATHDOC

\$SERVDOC: http://<server name>/ ; \$SERVCGI: http://<server name>/scripts

2) Mettre dans \$SYS: wwwxml.exe, fuzzy.cfg, wwwxml.cnf

wwwxml.cnf contient \$XML (path_inc("d:\inetpub\scripts\irdp\webexp\xml"))

3) Mettre dans \$APPLI: salle.jx, wxgoto.exe, webexp.cnf (contient l'ensemble des path et serveurs décrits ci-dessus), (wgestion.exe).

4) Mettre dans \$XML: les répertoires et fichiers XML avec à la racine des fichiers .js

5) Mettre \$DOC: les fichiers HTML avec à la racine ermitage.htm (et d'autres documents de base) qui contient:

```
var servRootDoc = "$SERVDOC/$PATHDOC" ;
```

```
var servRootCGI = "$SERVCGI/$PATHAPP" ;
```

```
var servRootXML = "$SERVCGI/$PATHAPP/xml" ;
```

6) Installer sous IIS le lien entre .xml et wwwxml.exe (PWS: modifier la base de registre de façon standard)

Remarques:

1) Les fichiers XSL contiennent l'adresse de \$servRootDoc

2) Les fichiers xml sont considérés comme des scripts

Apache / Linux 386 – NT?

1) Considérer les répertoires

\$APACHE: par exemple /usr/local/apache

\$APPLI: par exemple \$APACHE/share/cgi-bin/webexp/xml

\$PATHXML: webexp/xml ; \$PATHDOC: webexp/doc

\$ROOTDOC: \$APACHE/share/htdocs

\$XML: \$ROOTDOC/\$PATHXML ; \$DOC: \$ROOTDOC/\$PATHDOC

\$SERVDOC: http://<server name>/ ; \$SERVCGI: : http://<server name>/cgi-bin

2) Mettre wwwxml, wwwxml.cnf, fuzzy.cfg, wxgoto, salle.jx, (wgestion) dans \$APPLI (wwwxml.cnf contient: (path_inc("\$XML")), webexp.cnf contient l'ensemble des path et serveurs décrits ci-dessus (à regrouper)).

3) Paramétrer APACHE avec 'action' et addhandler' standard (lien entre .xml et wwwxml)

ermitage.htm dans \$DOC doit contenir:

```
var servRootDoc = "$SERVDOC/$PATHDOC" ;  
var servRootCGI = "$SERVCGI/$APPLI" ;  
var servRootXML = "$SERVDOCI/$PATHXML" ;
```

Remarques:

- 1) Les fichiers XSL contiennent l'adresse de \$servRootDoc
- 2) Les fichiers xml sont considérés comme documents

Annexe 8: Le journal

Ancêtres: ProfExpert, Utopia, et modules de transfert des données de ProfExpert en XML (SGML).

Dès l'automne 1999: Usage de Perl pour réaliser des parser XML. Réalisation de uxml / wwwxml puis wxgoto en Prolog.

Janvier 2000 : Publication de la première version du suivi : Reprise du projet ProfExpert, 11 janvier 2000.

A l'automne 2000 environ 200 fiches sont disponibles. L'ensemble fonctionne.

Noël 2000 : le nombre de fiches a doublé. Plusieurs pages sont interactives.

Février 2001 : présentation à la médiathèque du CPLN ; quelques ajustements pour la version MAC (à voir encore pour certaines fonctions qui renvoient sur des 'parent').

Mars 2001 : ajustement du moteur XSLT ; préparation d'une nouvelle tranche graphique ; préparation d'une applet pour la salle de la complexité ; travail aux salles de l'exposition Science et Cité.

Printemps 2001: test de la DTD et de quelques pages XML/XSL par C. Kottelat (supervision Luc R.). Pas trop mal.

Mai et été 2001: les concours fonctionnent.

Automne 2001: présentation au colloque RMT et aux journées de la formation prof (Lugano).

Hiver 2001: amélioration des xsl (notation vtml-1, qrep2-1, etc.) et transfert d'exercices de profexpert.

Aout 2002:

fonction ote(index, liste) (salle.jx)

newLoc est améliorée et peut être utilisée en interne avec des valeurs qui ne sont pas des chaînes.

fonction attribut_val_data(M,D): attribue aux variables contenues dans M les valeurs contenues dans D. Fait appel à attribut_val_basic(M,D) et à attribut_val_array(M,D)

Problème de valLoc lorsque la variable n'est pas définie, elle renvoie son nom (hésitation avec 'undefined' (undefined pas défini sous MSIE) ou "" (ce qui est le cas pour les variables globales)).

Réalisation de l'expertise pour la proportionalité.

Graphisme: onglet et fonds.

Septembre 2002

isBound(nvar) renvoie true ou false selon que la variable locale nvar est définie ou pas.

déclencheur 'onload' (voir sous qrep)

Automne 2002

Présentation dans le cadre de la formation FFA3, demande de réalisation d'exercices concernant les mélanges proportionnels.

Préparation d'activités 4-6 et rédaction de l'article pour STE.

Janvier-Février 2003

Introduction de « contextes » dans le cas de qrep2-2 (format libre) et de nouveaux éléments (NOTE, CARTOUCHE, BLOC, ...). Mise au point de la philosophie de l'attache systématique de fonctions JavaScript aux éléments XML définis.

Réalisation d'une synthèse de feuille de style qrep-qcm

Mars 2003

Le système de navigation fonctionne sous linux

Correction des lignes de status (ne se cachaient pas) à faire encore les corrections dans les salles (dépendent de salle.jx)

Introduction systématique de fonction makeXXX retournant des éléments HTML (simplification + évite des balises HTML dans des « string » jvs lorsqu'on construit des balises à partir de variables)

A voir pour une façon standard de fabriquer les éléments PEXP encapsulant le tout.

Avril - Mai 2003

Fabrication des « pistes matematica » avec certificat et enregistrement du score.

Cours HEP-BEJUNE (cf rapport Ismaël Ghodbane)

Introduction du « téléporteur ».

Introduction de la notion de groupe (cf inst.log sur machine Linux). Les groupes sont utilisés pour le suivi.

Les fichiers et modules accueil.htm, accueil2.htm, inscript.inc, util_file.pr*, prive.htm sont modifiés pour accueillir la notion de groupe.

Pour créer un groupe GR, il faut créer le fichier GR.cla dans les « data » et le repertoire GR dans /suivi. Il faut ajouter le nom du groupe dans GR.cla (opérations qu'il faudra intégrer à wxgestion).

Début d'installation de log (suivi). A mettre encore dans deplacement_salle, deplacement_partie etc. A corriger l'adresse de l'action dans les formulaires de suivi.

Juin 2003

Mise au point du format « flash » avec enregistrement des scores. Cet enregistrement est aussi effectué dans le module « proportionnalité ». Une fonction générale est définie pour cela : formEval.

Juillet 2003

Voir ate-div1 pour une interaction lacunaire relativement « sophistiquée ».

Démarrage sur une fenêtre sans menu (welcome.htm renommé welcom1.htm ; welcome.htm procède à l'ouverture d'une nouvelle fenêtre). (l'ancienne pourrait être effacée via un window.close()). Mais ne fonctionne pas partout pour des raisons de sécurité (anciennes versions, seules les fenêtres ouvertes avec window.open(...)) peuvent être fermées avec window.close(), les nouvelles versions autorisent également les premières fenêtres de l'history).

Novembre 2003

Introduction de suivi d'images à cliquer dans le concours hiver 2003-2004.

Mars 2004

Des zones et/ou des salles peuvent avoir un accès restreint à certains groupes. Les faits doivent être mis dans relation.pro.

zone_restricted_to(cadre, dom, sdom, groupe)

salle_restricted_to(cadre, dom, sdom, sujet, groupe)

Avril 2004

Modification de uxml pour pouvoir mettre les indications de sortie (html_output) dans la feuille de style xsl plutôt que dans le document xml.

Modification de wwwxml dans le même sens. Nouvelle variante de wwwxml (wwwxmlplus) permettant la compilation du fichier xml si pml n'existe pas.

Fabrication de xml2pml module « ligne de commande » pour faire les compilations (version windows et linux).

Test du dispositif d'accès restreint. Décision de rendre une salle à accès restreint dès qu'elle est dans une zone à accès restreint. Extension du dispositif au niveau de l'adjacence.

Examen de la possibilité d'améliorer le moteur XSLT notamment pour la prise en compte de Xpath du type `match= « name[att1='1' and att2='5'] »`

Refonte du « domaine » ATT_VAL. Cela rajoute un niveau. REQ prend son rôle et il devient soit un string ou une liste de ATT_VAL (ajout après parsing pour respecter l'ancienne philosophie), soit un PATH soit un PATH[REQ], soit un PATH/REQ.

REQ reprend le rôle de l'ancien ATT_VAL avec ajout des opérations booléennes ! (not) and et or. A voir si à la place l'usage d'un moteur existant sur le marché ne serait pas possible (XALAN).

Mai 2004

Passage de la DTD sous netBeans (mac)

Juin 2004

Ajout de `unLoc(nvar)` pour ôter une variable, `unLocList(lvar)` et `unLocAll()`. Utilisées pour corriger `qrep-exp` et ôter la variable locale « Pour1 » qui provoque des remplacements inopinés de valeurs de variables. A manier avec précaution. Milite pour faire des catégories de variables de fiches et leurs entours, variables de salles et variables de suivi.

Fabrication de `activite.xml`. A améliorer en fonction de nouvelles idées.

Fabrication du dispositif de « panneau » qui permet d'afficher des informations éphémères dans la zone de droite dans l'interface de navigation.

Juillet-Août 2004

Réalisation d'une zone dédiée aux révisions `hep-bejune`.

Correction : `unLoc` ajouté dans `proport.js` lors du passage d'un énoncé à l'autre. Correction de `proport.js` et `flash.js` pour qu'une « remise » ne soit faite qu'une seule fois par question (lorsqu'une reprise était mise à 3 par exemple, 2 erreurs successives provoquaient trois reprises ultérieures).

Octobre-novembre 2004

Ajout de style dans le modèle `qrep2-2.xml`. Fabrication d'un modèle fortement basé sur les éléments positionnés. (`situatio/sit-pg1.xml`). Amélioration en utilisant les contextes (`situatio/pyr-sou?.xml`).

Janvier 2005

Préparation de salles destinées à une expérimentation avec des étudiants de la HEP-BEJUNE.

Avril-mai 2005

Fabrication d'une version pour l'environnement « cygwin / Apache ». Il faut prendre la version Linux (qui utilise la variable d'environnement PATH_INFO à la place de comline pour la version IIS). Problème de transformation entre / et \ (path fichier et path url identiques sous Linux, pas sous cygwin). Actuellement, c'est fait de façon adhoc spécifique pour l'installation du système sous : c:\cygwin\var\www\htdocs\webexp\xml\ avec le symbole pathinfo pour compilation conditionnelle. A faire une version de traite_xml avec compilation séparée dirigée par des symboles linux et cygwin (par exemple). (et mettre path_root dans la version windows au cas de la compilation pour cygwin).

Octobre-novembre 2005

Mise à jour des salles concernant fonctions et relations (cours de math ESNIG). Avec des petits exercices de calcul de termes de suite ou de séries.

Réalisation d'activité pour recherche en collaboration avec HEP-BEJUNE. Modification du modèle qrep2-2.xsl pour pouvoir introduire plusieurs zones javascript (notamment sous forme de libraries « externes »). Utilisation d'une nouvelle librairie javascript pour dessiner des formes (voir la situation des pions sauteurs).

Mise au point des modèle touchpad et qcm-img3 en liaison avec école Corcelles-Cormondrèche.

Recompilation de wwwxml et uxml pour ajouter les fonctions comme valeur d'attribut.

Ajout du type de link bigPopup.

Janvier 2006

Salles probabilité et création du modèle qrep-mult. Réutilisation du modèle qrep2-2. Début de la création de bibliothèques javascript (arithm.js)

Avril-Mai 2006

Création de salles pour stat et prob. Avec notamment l'applet de simulation du lancer du dé.

Modification du modèle qrep-mult pour rendre les fichiers correspondants conforme à la DTD. Les fichiers correspondants, heureusement en petit nombre, sont migrés dans le nouveau format. Le problème de non conformité a été constaté lors de la tentative d'ajout de l'attribut SIZE à l'élément ITEM, alors qu'il était prévu d'insérer dans ITEM un FORM-REP (qui lui possède cet attribut). (de même, le groupe d'items font partie d'un éléments INTERACTION).

Mise au point laborieuse liée à des difficultés « théorique » concernant la définition de variable aléatoire et du lien avec caractère et série statistique. Un fichier pdf est joint comme aide qui donne la tabulation de la fonction Phi.

Juillet 2006

Programmation (javascript) de memories et d'un puzzle en vue de créer un modèle. Introduction d'un nouveau type de contexte : interface. Cela semble le bon niveau : Les attributs sont plutôt devenus des métadonnées de situation du document, les variables (LOC) plutôt liées au contenu.

Examen des possibilités (qui existent) de contrôler du son en jvs (en vue aussi du projet EOLE).

Août 2006

Ajout d'exercices pour CPI-FR avec usage des modèles existant. Un usage intéressant du modèle qrep2-2 est fait avec utilisation de BLOC. Une amélioration est apportée à la feuille de style pour pouvoir mettre un BORD aux blocs de type « table » (marche parce que seule la première règle qui match est prise. Sinon, il faudra utiliser la valeur « ligne » de DISPO explicitement. Dans ce cas intinerai/nu_pb4.xml serait à corriger) (voir \$xml/problem1/cpi-exe-int03.xml).

Septembre 2006

\$xml/util/suivi-ok et \$xml/util/suivi-nok.xml améliore les documents de feed-back msg-ok et msg-nok (ajoute le suivi pour ok et permette des messages différenciés pour nok). (voir \$xml/calcul2/cpi-exe-cal01.xml)

Janvier 2007

Introduction du modèle flash pour les défis.

Août 2007

Modèle qtml-2 et correction de NOTE dans vtml-2x.xml, dftml.xml, qrep-qcm.xml. Usage de ce modèle dans \$xml/mathbas2/th-den01.xml (lemme de Bernouilli).

Décembre 2007

Pour l'élément VAL-VAR défini dans qrep2-2.xml ajout des FORMAT fix2 (écriture au centième) et fix05 (au 5 centième). Ces formats utilisent les fonctions (dans salle.jx) format100 et format05 (nouvelle) (voir cpi/int-jour1.xml).

Mis dans \$doc/util tab-propXY.htm qui permet d'afficher des tableaux à 4 cases. Un compromis a été adopté d'afficher les nombres avec 3 chiffres après la virgule (voir cpi/int-jour1.xml).

Février-Mars 2008

Salle cpi/numération : Installation d'une salle « numération » dans la zone cpi. Reprise des exercices de la salle « lire et écrire des nombres ». Ajout d'attributs « copyright » pour faire nettoyer la zone de note dans les enchaînements.

Salle cpi/monnaie : Création d'exercice sur la monnaie (salle « monnaie » dans la zone cpi). A cette occasion un nouvel contexte expert de scheme 'unite' et valeur 'monnaie' a été utilisé. Il est utilisé dans la bibliothèque flash.js pour l'affichage des énoncés (4.50 et non 4.5) (voir \$xml/cpi/exe-mon07.xml).

Création d'exercices sur la monnaie (restitution) (\$xml/cpi/exe-mon0[1-9].xml et \$html/cpi/hlp-mon1.htm).

Salle cpi/calcul : Création d'un exercice de calcul flashé à perpète pour la multiplication (ajout de la prise en compte du s_paramètre naturel/multiplication dans flash.js). A voir pour essayer de limiter $1*0$, etc. Ajout de deux exercices flashés (10 items chacun) de multiplications simples ($3 * 34$; $3*123$) (\$xml/cpi/exe-mu0[1-3]).

Salle cpi/mesure : ajout de tableaux à compléter sur les longueurs (\$xml/cpi/exelon0[1-4]) aide associée hlp-long.htm. Idem pour les capacités et les masses.

Modification de qrep-mult.xsl pour pouvoir ajouter des cases vides au tableau (<NULL/>).

En principe le suivi est assuré par \$xml/util/suivi-ok.xml ou \$htm/util/suivi-ok.htm. Parfois, la réponse juste appelle par le link une page de feed-back ad hoc. Pour assurer le suivi on peut mettre dans cette page l'appel à formEval grâce à un élément VAL-VAR de type 'fct' (cas xml, voir \$xml/itinerai/nu_ak1a.xml) ou un appel direct (cas htm). Ajout du suivi aux exercices de numération de \$xml/itinerai.

Mars-avril

Migration de toute une série d'éléments théoriques de Prof'Expert à propos de l'écriture des nombres.

Création du modèle touchpad2 qui reprend touchpad en utilisant le cumul. C'est-à-dire que le clic sur une image ajoute la valeur associée à la valeur précédente (voir \$xml/cpi/exe-monp10.xml). Les deux modèles utilisent touchpad.js, le premier avec interaction/ mode map/simple et l'autre map/cumul. Ce format ajoute les attributs permettant de fixer la grandeur des zones d'interaction.

L'occasion de corriger quelques bugs de touchpad.js (liés au fait que les nouvelles versions de Javascript lève une erreur pour un objet xx.yy non défini, au lieu de mettre la valeur undefined, une erreur provenait de l'usage de nb-exe, nombre d'exercices à réaliser qui augmente lié à la remise alors que le nombre d'exercices disponible reste le même).

Annexe 9: Les fonctions à disposition dans une salle

Ce sont les fonctions qui peuvent être appelées à partir d'une page html et/ou xml. Certaines de ces fonctions ont des appels prédéfinis depuis des pages xml, au chargement ou encapsulés dans des balise XML.

Affichages dans des fenêtres

afficheNote(msg), clearNote(),
afficheLiens(msg), clearLien(),
afficheTexte(msg)
affichePopUp(msg)
hidepic() (ferme la fenêtre popup)
afficheCartouche(evt,ptrLayer,ptexte) affiche ptexte dans une petite bulle. Si ptexte est précédé de :, il s'agit d'une variable (à améliorer la présentation)
afficheCartouche0 et afficheCartouche1 affiche un texte contenu dans un fichier (ev. via un lien LNK).
cacheCartouche(lay)

afficheRef(ref,cop,from) (fait le 'onload' à la suite, efface la fenêtre note si pas de référence)
afficheRef2(ref,cop,from) (ne fait pas le onload, n'efface pas la fenêtre)
afficheIntro() (si un lien onload existe)

Fabrication d'éléments

makeImageEl(url,width,height,center) (-1,-1 si w et h inconnu, 1 = centré)
(à modifier ev makeImageEl(url,"width,height,center") le paramètre étant fournit par l'attribut PARAM)
makeInput(type,name,value)
makeInput1(type,name,size)
makeInput2(type,name,size,onchange)
makeButton(name,value,style,onclick) (à verifier si transféré sur v. w32)
makeTR(nbr_cell,tab_cell)
(peuvent s'utiliser dans un document.write, lui-même inclus dans une balise. Pourraient être encapsulées dans des éléments XML).
makeTag(tag) retourne un tag
makeTagAtt(tag,att) retourne un tag avec la liste d'attributs
makeEndTag(tag)
makeSimpleEl(tag,str)
makeRespImage(img,type,scheme,border,reponse) retourne une image liée à simpleReponse (à mettre ultérieurement dans un élément ad hoc)

afficheDirect(idref) (affiche selon les directives de la balise link avec IDRE='idref')
afficheDirectByIndex(index) (idem à partir de l'indice dans la table des liens)
afficheIntNote(index) (identique à la précédente ?)
afficheRoomCtxt() (appelée sur 'ONLOAD' de next.htm. Provoque l'affichage des frames navi et next)
afficheFrise(img,sty,text) (affiche la frise, à inclure bold dans le style !)


```
afficheActAdj() (affiche dans next)
afficheAdj() (affichage dans navi (salles adjacente et sortie))
```

```
// *** affichage dans la zone link
// information provenant du fichier XML (voir HTML !)
// via le tableau wxTabLink
// il y a changement de salle si l'activite n'est pas
// dans la salle et si la fonction est precedee par '@'
// on pourra ajouter recherche d'une autre salle si url="" et name <>""
// sinon erreur !
```

```
function afficheLink()
function afficheNext(elLink, fonction)
```

Système de variables

```
function valLoc (nvar)
function vL (nvar)
function isBound(nvar)
function newLoc (nvar, val) {
function unLoc(nvar)
function char_rnd(str)
function valVar(nvar)
```

```
function tab_rnd(tab) {
function rnd(n) {
function ote(index, liste)
function ote_el(el, liste)
```

note : attention les listes sont passées par leur adresse. De façon générale les fonctions prenant des listes simulent un passage par valeur, c'est-à-dire que les listes sont reconstituées dans la fonction pour être retournée ensuite.

```
function attrib(n, str, lloc) // attribue la valeur str a n variable locales
function attribut_val_data(M, D) // attribue aux elements d'un modele M, les valeurs
de D
function attribut_val_basic(M, D) {
function attribut_val_array(M, D) {
```

```
// pour systeme de correction
```

```
// sortie (pour lier par exemple a IDREF="exit")
// peut aussi être utilise tout seul !
```

```
function sortie()
```

```
// lancement du controle
// variable globale a la salle (plutot que valLoc !)
```

```
var locReponse = "" ;
```

```
// *****
```

```
function controleReponse(formulaire) {
// comparaison modele et reponse actuelle selon type et scheme

// sans formulaire
function simpleReponse(type,scheme,rep)
function formEval(text) fonction qui fournit un formulaire pour le suivi (qui
enregistrera text)
function afficheFormEval(msg, text) affiche le message msg dans la zone de note
et le formulaire du suivi.

// utilitaires

function isInteger(n)
function round100(r)
function format100(r)
function format05(r) (12.07)
function puis(a,n)
```

Annexe 10: Idées et à faire

- 1) Faire un link de type ONLOAD, s'il existe il se fait au chargement de la page ! (fait septembre 2002, ce n'est pas le type mais idref qui est "onload")
- 2) A documenter: Layer cartouche <DIV ID="cartouche"> avec le style cartoucheDIV et cartoucheA (à voir év. pour rassembler le style cartoucheDIV dans le même objet)
- 3) Les appels de fonctions : parent.afficheCartouche() (si texte commence par ':' -> recherche de variable (cf mathbas0/ui_785.htm) et parent.cacheCartouche sont à mettre dans une balise A (de class cartoucheA) sur OnMouseOver et OnMouseOut (voir rallye/envoi-1.xml). Amélioration apportée voir problem1/ui_1898.xml et les aides associées. Introduction sous forme d'élément (voir vtml2-2.xsl).
- 4) A faire quelque chose pour compatibilité entre les navigateurs et à utiliser des techniques semblables aux layers 'enonce', 'consigne', formulaire. Cartouche ne fonctionne pas avec MSIE, enonce & co ne fonctionne pas avec NST 4 et 6. (ok pour MSIE et NST 4, 09.02)
- 5) Voir le rapport entre les valLoc et des variables standard définies dans salle.jx (comme par exemple reponse, ...). Plutôt que <LOC NAME="nrep" VALUE="0"/> on pourrait avoir une zone script avec parent.nrep = 0 (nrep aurait été déclarée dans salle.jx à nommer ev. locNrep). Fait en partie (02.03), voir qrep2-2.xsl.
- 6) Dans la zone de l'image faire un appel à un cgi qui choisirait un texte, une icône, etc. en fonction du lieu, usager, etc. (fait en partie juin 2004 sous le terme de dispositif de panneau).
- 7) Erreurs dans les FUNCTION de liens (parfois fiche à la place de info, exemples à la place de exemple, etc.) une correction est apportée sur le nom des fichiers des pictos.
- 8) Attention : dans webmath.lex mettre les doublures par ordre décroissant du « level » (à rendre l'ordre indifférent).
- 9) FORM-REP à introduire SELECT
- 10) Faire un modèle pour les "itinéraires" (cf itinerai/md_df0.xml). (en partie réalisé avec vtml-1d.xsl et dftml.xsl, mars 2003 cf nu_df0.xml, nu_df1.xml, nu_adf1.xml, à mettre tout avec des variables javascript stables lcXXX à la place de newLoc)
- 11) Correction des lignes de status dans les salles (dépendent de salle.jx)
- 12) A voir pour une façon standard de fabriquer les éléments PEXP encapsulant les fonctions makeXXX.
- 13) L'appel à afficheDirect ne permet pas de changer de salle (lorsque le lien est avec @ devant la fonction). A voir si c'est possible de rétablir. Ou sinon à faire lors de l'introduction directe de <A IDREF ...>

- 14) A mettre la feuille de style en attribut.
- 15) itinerai/md_pb8.xml et itinerai/nu_df3.xml introduisent une nouvelle interaction. A voir dans le contexte pour indiquer si le temps est pénalisant ou pas !
- 16) initialisation systématique de rep_no et poids_init dans qrep2-2.xml. A refaire une systématique pour les interactions (cf itinerai/nu_df3.xml pour les questions posées). Notamment problème des listes d'items qcm et les listes qui défilent (avec des input de type text ou des layers ...).
- 17) qrep-fls.xml implémente la zone de texte sensible par « return ». A mettre aussi dans les formulaires des concours ...
- 18) L'ordre aléatoire implémenté dans flash.js, à mettre dans les autres (propor.js, etc.)
- 19) Attention : Le contexte « tolerance » était sous interaction dans PE, il est sous contrôle
- 20) Les réponses (qrep2-2) à prendre comme texte sont actuellement dans les défauts à faire éventuellement un type "texte"
- 21) Mettre le package à disposition comme variable globale (wx.Package)
- 22) Intégrer la création de groupes à wxgestion.
- 23) Insérer le traçage (log - suivi) dans déplacement_salle, déplacement_partie etc. Corriger l'adresse de l'action dans les formulaires de suivi. (?)
- 24) Compléter le modèle lacune avec la fabrication du suivi (appeler depuis l'applet une fonction du type afficheFormEval(str) (en cours de réalisation mai 04)
- 25) Revoir l'élément INTLNK pour pouvoir ajouter des balises (de plus à vérifier le comportement du saut de ligne (\n) selon le système).
- 26) Faire une sorte de lien standard entre une fonction javascript Z et un élément XML (<Z ou <JVS f="Z" param="...">). C'est peut-être déjà fait avec VAL-VAR à voir.
- 27) Etendre le modèle qsl2-2 pour intégrer les éléments dynamiques (cd situatio/pyr-sou?.xml).
- 28) Attention : Le contexte « remise » a été mis parfois sous interaction, il est à mettre sous contrôle (proport.js utilise les deux pour assurer la compatibilité, flash.js est juste)
- 29) Permettre des INTLNK plus sophistiqués en stockant les infos sous la forme de « layers » plutôt que comme paramètre d'une fonction.

30) Refaire les objets et fonctions liées à « cartouche » en tenant compte du DOM (notamment ne plus utiliser un <A> comme ancre mais un , par exemple). Revoir aussi le positionnement (attention à la compatibilité ascendante, les tableaux de la proportionnalité, par exemple ui_785.htm, utilise le fait que 2 layers sont imbriqués : les coordonnées de l'événement du layer intérieur sont utilisées pour positionner un objet à l'extérieur).

31) Documenter et rassembler les sources (voir pour passage sous 5.2).

32) Faire des exercices pour des fonctions affines en utilisant partiellement les propriétés des tableaux proportionnels.

33) L'élément non standard <xsl:include-txt ..., surtout utilisé pour inclure du code javascript, pourrait être remplacé par un élément <LINK ... dans le fichier HTML produit.

34) Fabrication de memory/3, etc. à partir de retouches mineures apportées à `puzzle.js` (à l'aide de `parseInt(lcInteraction.type.charAt(lcInteraction.type.indexOf(« / »)+1))` on récupère le mode.

35) A mettre à jour l'élément BLOC de `dftml.xsl`, `qrep-qcm.xsl`, `vtml-2.xsl`, `vtml-2p.xsl`, `vtml-2s.xsl` selon modèle `qre2-2.xsl`. (ok, 07.09.06) Vérifier que de façon standard, dans un tel cas (si BLOC possède à la fois l'attribut DISPO et BORD) seule la première règle est activée.

36) L'appel du certificat se fait à travers un script dont l'url est codé en dur dans la page d'appel. A introduire la variable globale correspondante (fichier à corriger `$xml/itinerar/*_df-e.xml`, `$xml/concours/envoi*`, `$xml/rallye/envoi*`). (ok, 24.09.06)

37) Ajouter une option à l'interaction flash pour générer des suites de calculs de longueur fixée (comme pour perpète mais fini :). C'est le contexte contrôle/remise qui est utilisé pour perpète. A voir comment arranger la chose.

38) Ajouter dans les modèles (notamment `qrep-exp.xsl`), les différents formats à l'élément VAL-VAR.

39) Faire la liste des contextes actuellement utilisés (à mettre à jour l'annexe 12 mise à jour).

40) La fonction `afficheFormEval(msg, txt)` est utilisée dans `suivi_ok` pour permettre l'enregistrement (optionnel) du suivi. A installer ce dispositif de suivi lorsque la quittance est réalisée par une page xml (à introduire la balise idoine dans les modèles, notamment en utilisant le modèle `vtml-2p.xsl`, par exemple). (voir 26) (ok, 8.03.2008, l'éléments VAL-VAR de type 'fct' permet de faire la chose. Voir la nouvelle version de `$xml/itinerai/nu_ak1a.xml`).

41) Ajouter la possibilité dans le contexte parametre (`flash.js`) non seulement la valeur maximale des nombres à créer (perpète) mais également le minimum (utile pour éviter de proposer $0 * 1$, $0 + 45$, etc.).

42) Améliorer la lisibilité du suivi (problème d'accentuation, alignement, grandeur de la fenêtre, etc.)

43) A propos des exercices de flash (par exemple \$xml/itinerai/md_pb8.xml). Il serait utile de cacher les scripts dans un modèle existant (qrep-qcm, par exemple) et de se limiter à liste questions et items. Une simple modification du contexte permettrait alors de présenter les questions sous forme d'une liste visible totalement ou sous la forme d'une séquence (à temps limité ou non). A remplacer les zones de texte qui présentent les données par des « layers ».

44) Permettre un défaut aux attributs top-msg, top-con, height-zone dans touchpad2. Ajouter les autres paramètres dans les attributs.

Annexe 11: code d'une "piste mathematica"

```
<?XML VERSION="1.0"?>
<!DOCTYPE PEXP SYSTEM "pexp.dtd">
<?XML-stylesheet TYPE="text/xsl" HREF="vtml-1d.xsl"?>
<?XSL-FLAG VALUE="html-output"?>

<PEXP>

<HEAD>
<TITLE>Introduction</TITLE>
<ATTLIST>
<ATT NAME="base" VALUE="itinerai"/>
<ATT NAME="name" VALUE="index-nu-df"/>
<ATT NAME="file" VALUE="nu_df0.xml"/>
<ATT NAME="dtype" VALUE="fiche"/>
<ATT NAME="copyright" VALUE="(c) AM & LOP, 2003"/>
<ATT NAME="reference" VALUE=""/>
</ATTLIST>

<LINKS>
<LNK IDREF="debut" NAME="@nu_df1" HREF="itinerai/nu_df1.xml"
FUNCTION="question" TYPE="texte" TITLE="Commencer"/>
<LNK NAME="RMT" TITLE="Rallye mathématique transalpin" TYPE="site"
TARGET="extern" HREF="http://www.rmt-sr.ch" FUNCTION="info"/>
<LNK NAME="Math-Ecole" TITLE="Revue Math-Ecole" TYPE="site" TARGET="extern"
HREF="http://www.math-ecole.ch" FUNCTION="info"/>
<LNK NAME="SENS" TITLE="Société des enseignants
neuchâtois de sciences" TYPE="site" TARGET="extern"
HREF="http://www.abord-ch.org/sens/" FUNCTION="info"/>
<LNK NAME="ABORD" TITLE="Association pour les apprentissages de base avec
l'ordinateur" TYPE="site" TARGET="extern" HREF="http://www.abord-ch.org/"
FUNCTION="info"/>
</LINKS>

</HEAD>

<BODY>
<H3>Bonjour</H3>

<P>Ce défi te conduira vers un diplôme en lecture et
écriture des nombres</I>
</P>

<P>Tu auras 5 situations à résoudre.</P>

<P>A la fin du parcours, tu recevras un certificat.</P>

<P>A tout moment tu peux abandonner pour revenir dans la salle
d'exercices.</P>

<P>Tu pars avec un crédit de 10 points. A chaque erreur tu perds des
points.</P>

<P>Pour commencer, clique sur commencer !</P>

</BODY>
</PEXP>

<?XML VERSION="1.0"?>
<!DOCTYPE PEXP SYSTEM "pexp.dtd">
<?XML-stylesheet TYPE="text/xsl" HREF="qrep-qcm.xsl"?>
```

```

<?XSL-FLAG VALUE="html-output"?>

<PEXP>

<HEAD>
<TITLE>En chiffres</TITLE>
<ATTLIST>
<ATT NAME="base" VALUE="itinerai"/>
<ATT NAME="name" VALUE="nu_df1"/>
<ATT NAME="file" VALUE="nu_df1.xml"/>
<ATT NAME="dtype" VALUE="exercice"/>
<ATT NAME="copyright" VALUE="(c) AM & LOP 2002"/>
<ATT NAME="reference" VALUE=""/>
</ATTLIST>

<CTXTLIST>
<CTXT NAME="interaction" SCHEME="type" VALUE="qrep"/>
<CTXT NAME="interaction" SCHEME="mode" VALUE="qcm/radio"/>
<CTXT NAME="interaction" SCHEME="reprise" VALUE="avec"/>
<CTXT NAME="interaction" SCHEME="nbReprise" VALUE="3"/>
<CTXT NAME="controle" SCHEME="poids" VALUE="3"/>
<CTXT NAME="controle" SCHEME="penalite" VALUE="1"/>
</CTXTLIST>

<LINKS>
<LNK IDREF="link" HREF="itinerai/nu_adf1.xml" FUNCTION="feed-back"
TYPE="note" TITLE="Feed-back 1"/>
<LNK IDREF="exit" HREF="itinerai/nu_out.xml" FUNCTION="feed-back"
TYPE="texte" TITLE="Sortie"/>
<LNK IDREF="msg" HREF="itinerai/df-msg.htm" FUNCTION="message"
TYPE="note" TITLE="Nb d'essais"/>
<LNK IDREF="onload" HREF="itinerai/df-imsi.htm" FUNCTION="message"
TYPE="note" TITLE="Nb d'essais"/>
</LINKS>

</HEAD>

<CONTENT>
<INTRO>
<P>Clique chaque fois sur le nombre en chiffres correspondant
au nombre en lettres.</P>
</INTRO>

<ITEM NAME="1">
<ENONCE>mille trois cent cinquante trois</ENONCE>
<CHOICE>130053</CHOICE>
<CHOICE>1300353</CHOICE>
<CHOICE STATE="juste">1353</CHOICE>
<CHOICE>1335</CHOICE>
</ITEM>

<ITEM NAME="2">
<ENONCE>dix mille quatre-vingt un</ENONCE>
<CHOICE>10421</CHOICE>
<CHOICE>1081</CHOICE>
<CHOICE>100004201</CHOICE>
<CHOICE STATE="juste">10081</CHOICE>
<CHOICE>104201</CHOICE>
</ITEM>

<ITEM NAME="3">
<ENONCE>huit cent nonante sept</ENONCE>
<CHOICE>80097</CHOICE>

```



```

<CHOICE STATE="juste">897</CHOICE>
<CHOICE>800907</CHOICE>
</ITEM>

</CONTENT>
</PEXP>

<?XML VERSION="1.0"?>
<!DOCTYPE PEXP SYSTEM "pexp.dtd">
<?XML-STYLESHEET TYPE="text/xsl" HREF="dftml.xsl"?>
<?XSL-FLAG VALUE="html-output"?>

<PEXP>

<HEAD>
<TITLE>En chiffres</TITLE>
<ATTLIST>
<ATT NAME="base" VALUE="itinerai"/>
<ATT NAME="name" VALUE="nu_adf1"/>
<ATT NAME="file" VALUE="nu_adf1.xml"/>
<ATT NAME="dtype" VALUE="fiche"/>
</ATTLIST>

<LINKS>
<LNK IDREF="suivant" NAME="@nu_df2" HREF="itinerai/nu_df2.xml"
FUNCTION="question" TYPE="texte" TITLE="Question suivante"/>
</LINKS>
</HEAD>

<CONTENT>
<INTRO>
<P>La r&eacute;ponse <VAL-VAR NAME="#locReponse"/> est juste bravo !</P>
<P><VAL-VAR NAME="#bilan"/></P>
<P>Pour la question suivante, clique sur <SPAN STYLE="font-weight:
bold"><NOTE IDREF="suivant">question suivante</NOTE></SPAN> !</P>
</INTRO>
</CONTENT>
</PEXP>

```

Annexe 12 : DTD (pexp3.dtd)

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- *****

Project code: Web'math (c) ABORD, 1999
Object: XML DTD
File: pexp3.dtd
Usage: Raw data description
Version: from 1.0
Dated: june 1999
Modified: may 2000, september 2000, january 2001, april 2001, april 2002,
         february 2003
* Links can be directly used in the document without LINKS
  declaration
* LINKS element is integrated in HEAD element
* internal links (notes, questions) are integrated in "BODY"
  element.
* January 2002, cleaning
* April 2002, new version 1.1 not totally compatible with 1.0
* Feb 2003, extension of element VAL-VAR, NOTE, CARTOUCHE
* May 2006, cleaning

Note: the file with the leading number is the reference.
Examples can be found in the old version in /PROLOG/PEXP_MIG directory.
See also the files pexp.dtd and pexp<n>.dtd for further comments,
mapping with the old Prof'Expert format (color, etc.).

Warning: to synchronize with the linux & windows version
***** -->

<!-- *** Entities for basic elements *** -->

<!ENTITY % elem.mini "#PCDATA | IMG | A">
<!ENTITY % elem.base "#PCDATA | FORMULE | BBOARD | PHRASE | IMG | A | DIV">
<!-- Note: en principe de type text avec BBOARD qui peut devenir bloc et
      DIV qui contient du (X)HTML ! -->
<!ENTITY % elem.plus "%elem.base; | VAL-VAR | VAL-CHAMP | NOTE">
<!ENTITY % elem.spec "INPUT">

<!-- ***** -->

<!-- *** Global document *** -->

<!ELEMENT PEXP (HEAD, (BODY | CONTENT))>
<!-- %content.pexp; -->

<!-- ***** -->

<!-- *** Header *** -->

<!ELEMENT HEAD (TITLE, ATTLIST?, CTXTLIST?, VARLIST?, SCRIPT?, LINKS?)>
<!ELEMENT TITLE (#PCDATA)>

<!-- *** Attributes *** -->

<!ENTITY % att.didactique "cadre|domaine|sdomaine|sujet|notions
      |dtype|keywords|level">
<!-- attention: notions se rapporte a des mots notionnels, keywords
      se refere au contenu (a corriger les documents RMT) -->
<!-- a voir encore le probleme du lexique -->
<!ENTITY % att.technique "date|nodenr|name|base|variable|file
```

```

|ttype|window">
<!ENTITY % att.administratif "from|copyright|reference|langue|auteur">

<!ELEMENT ATTLIST (ATT)*>
<!ELEMENT ATT EMPTY>
<!ATTLIST ATT
    NAME (%att.didactique;|%att.technique;|
          %att.administratif;) #REQUIRED
    VALUE CDATA #IMPLIED
    SCHEME CDATA #IMPLIED>

<!-- Quand plusieurs "values" sont autorisees (liste), elles sont separees
    par des espaces (dans Prof'Expert, il y avait des ;). A typer
    ulterieurement par NMTOKENS

```

* didactique

```

cadre : histoire* (culture), [hist]
       vie-courante (applications), [appl]
       fiction (enigme), [situ]
       jeu* [jeux]
       theorie* (approche formelle) [theo]
       informatique*, [info]
       didactique (informations didactiques), [dida]
       enseignement*(expose didactique, exercice), [ense]
       none

```

note: * marque les cadres repris comme interet de visite (??)
entre [] le nom dans Ermitage

level : 1 (cp) 2 (ce) 3 (cm) 4 (sec 6-7) 5 (sec 8-9) 6 (ly)
7 (specialiste) 8 (didacticien)

note: ce level est attribue lors de l'inscription, mais
tous ne sont pas libres de choix !
Lors de l'inscription existent encore:
0 (visiteur) 9 (administrateurs)
si le level est "" (null) on garde l'ancien

note: a voir pour dehierarchiser

domaine : arithmetique, algebre, geometrie, ensemble-relation,
logique, analyse, combinatoire, statistique,
algorithmique, topologie, general

note: peut dependre du cadre et du niveau

sous-domaine : depend du domaine

sujet : depend du sous-domaine

Du point de vue "museographique" le domaine correspond Ω l'etage et
le cadre a une "aile" du batiment. Des zones plus
petites correspondent a un sdomaine (groupes de salles),
un sujet est une salle.

notions : liste de notions mathematiques
dtype : fiche, situation, probleme, exercice, atelier, corrige
note: a completer depend du domaine et du cadre
keywords: liste liee au contenu (habillage)
level : niveau auquel s'adresse la fiche

note: actuellement keywords a ete aussi utilise pour 'notion'

* technique

```

date      : (SCHEME="std | old") (std = default)
nodenr   : ancienne numerotation pourra etre remplace par name
name     : (note: l'ancien printname est devenu TITLE)
base     : anciennement le fichier hba/ actuellement le directory
file     : le nom du fichier
window   : reprend les informations de prof'expert sur la fenetre; a
          voir le statut actuel (elements de style ??)
ttype    : prevu pour differencier fiche (xml, html), interface, menu,
etc. (pas utilise)
langue   : par default fr
variable: abandonne au profit des variables a part entiere (VARLIST)

note : window : (avec SCHEME="old") WC/FC/TOP/LEFT/HEIGHT/WIDTH
      WC mod 16 = couleur caractere ; WC div 16 = couleur fond
      idem pour FC (FrameColor). Voir les anciens fichiers pour des
      precisions.

```

* administratif

```

from      : pour indiquer la provenance (notamment migration prof'expert)
          avec l'idee de "transmis par" dans le cas ou ce n'est pas
          l'auteur.
copyright : detenteur du copyright
auteur    : l'auteur (les leres fiches -> aout 2001 n'ont que le
          copyright)
reference  : litterature (;-liste)
langue    : langue utilisee selon la denomination standard (fr, en, it)
-->

```

```
<!-- ***** -->
```

```
<!-- *** Contexts *** -->
```

```
<!ELEMENT CTXTLIST (CTXT*)>
```

```
<!ELEMENT CTXT EMPTY>
```

```
<!ATTLIST CTXT
```

```

  NAME (expert|controle|interaction) #REQUIRED
  VALUE CDATA #IMPLIED
  SCHEME CDATA #IMPLIED>

```

```
<!--
```

```
<CTXT NAME="controle" SCHEME="poids" VALUE="3"/>
```

```
<CTXT NAME="controle" SCHEME="penalite" VALUE="1"/>
```

NAME	SCHEME	VALUE
expert	dom	none (default) ; calcul
expert	psy	
expert	did	lacune ; calcul
expert	mode	proportionnalite
expert	unite	monnaie

controle	tolerance	none (default) (textuel)
controle	tolerance	calcul (1+1 est admis pour 2)
controle	remise	oui ; non
controle	ordre	original ; fixe ; aleatoire
controle	poids	N
controle	penalite	N
contrôle	parametre	n,max (nb de termes, val. max)
contrôle	s_parametre	naturel/addition, etc.

interaction	type	libre

```

interaction |      type      |      lacune
interaction |      mode      |      crochet ; enigme ; choix
*****
interaction |      type      |      qrep
interaction |      mode      |      normal ; qcm ; qcm/radio
interaction |      feed-back |      oui ; non
interaction |      reprise   |      oui ; non
interaction |      nbReprise |      N ; infini
interaction |      choix     |      N
interaction |      tempo     |      N
*****

```

A completer et a voir encore: ton, sexe, option imposee ou non, (mais le menu Ω disposition pourrait suffire)
-->

```
<!-- ***** -->
```

```
<!-- *** Variables *** -->
```

```
<!ENTITY % var.element "VAR | LOC">
```

```
<!ELEMENT VARLIST (%var.element;)+>
```

```
<!-- variables "globales" -->
```

```
<!ELEMENT VAR EMPTY>
```

```
<!ATTLIST VAR
```

```
    NAME CDATA #REQUIRED
```

```
    VALUE CDATA #REQUIRED>
```

```
<!-- variables "locales" (liee a une salle) -->
```

```
<!ELEMENT LOC EMPTY>
```

```
<!ATTLIST LOC
```

```
    NAME CDATA #REQUIRED
```

```
    VALUE CDATA #REQUIRED>
```

```
<!-- ***** -->
```

```
<!-- *** Scripts *** -->
```

```
<!ELEMENT SCRIPT (#PCDATA)>
```

```
<!ATTLIST SCRIPT
```

```
    LANGUAGE CDATA #REQUIRED>
```

```
<!-- a voir la liaison avec LINKS -->
```

```
<!-- a considerer le <![CDATA[ ... ]]>-->
```

```
<!-- ***** -->
```

```
<!-- *** Links *** -->
```

```
<!ENTITY % link.element "LNK | INTLNK">
```

```
<!ENTITY % type "texte|note|lexique|popup|bigPopup">
```

```
<!ENTITY % function "reference|aide|coup-pouce|relance|solution|
info|message|feed-back|outil|exercice|defi|
hlp-guide|exemple|methode|analyse|next|
curiosite|question|objectif">
```

```
<!-- attention a verifier; curiosite question avaient ete mis
sous type. -->
```

```
<!-- liens sur style et/ou script. Est-ce utile ? -->
```

```
<!ELEMENT LINKS (%link.element;)+>
```

```
<!ELEMENT INTLNK (%elem.base;)*>
```

```

<!ELEMENT LNK EMPTY>
<!ATTLIST LNK
  IDREF CDATA #IMPLIED
  HREF CDATA #REQUIRED
  LEVEL CDATA #IMPLIED
  NAME CDATA #IMPLIED
  SEGMENT CDATA #IMPLIED
  TARGET CDATA #IMPLIED
  TITLE CDATA #IMPLIED
  TYPE (%type;) "texte"
  FUNCTION (%function;) #IMPLIED>

<!-- Warning: IDREF reserve !! -->
<!-- TARGET fenetre, dont extern (ouvre nouvelle fenetre). Utilisation
a eviter si possible (type & fonction permettent souvent
de calculer la target

IDREF si il est non vide cela indique qu'un lien est fait depuis le
le document. Donc le lien ne sera pas affiche dans une partie
next ou lien d'une salle (voir aussi "WEBMATH-History.doc" pour
les details)

NAME nom du document complete par SEGMENT (ancre)
-->

<!ATTLIST INTLNK
  IDREF CDATA #IMPLIED
  LEVEL CDATA #IMPLIED
  TARGET CDATA #IMPLIED
  TITLE CDATA #IMPLIED
  TYPE (%type;) "note"
  FUNCTION (%function;) #IMPLIED>

<!-- du point de vue jvs, les donnees de LNK et INTLNK sont transmises
de la meme facon. Dans le cas de INTLNK le contenu va dans le
slot reserve a HREF. Un slot 'place' dans les donnees marque
la difference
-->

<!-- Remarques

1) info = reference (a preferer info)
2) lexique: a voir si ce n'est pas une option toujours presente ?
3) methode: analyse didactique pour apprenant (y compris objectif)
4) analyse: analyse didactique pour le formateur

5) A voir lien communication (forum etc.), et ev. d'autres informations
sur le lien (langue). Il y aura aussi des 'donnees complementaires
(regle, switch, etc. pour le guide, par exemple) a moins
que cela ne soit insere dans les data

6) A voir aussi l'idee de la "methode" qui permet de suivre l'info
(fichier html, xml, autres techniques ssi)

TYPE: donne le type d'objet lie et/ou la cible (sera a completer)

FUNCTION: donne une indication d'usage
Quelques fonctions peuvent avoir un TYPE par default
-->

<!-- ***** -->

```

```

<!-- *** Content *** -->

<!ENTITY % content.pexp "(BODY | CONTENT)">

<!ENTITY % elem.comp "INTRO | PROPOSITION | EXPRESSION | EXEMPLE |
DEFINITION |
                CITATION">

<!-- autres elements: CONSIGNE, ENONCE, FORM-REP -->

<!ENTITY % body.content "ANY"> <!-- (X)HTML -->
<!ELEMENT BODY %body.content;>
<!ATTLIST BODY
        ONFOCUS CDATA #IMPLIED
        ONLOAD CDATA #IMPLIED
        ONUNLOAD CDATA #IMPLIED>
<!-- futur %body.content; : jeu XHTML reduit a decider -->

<!ELEMENT DIV %body.content;>
<!-- attribut standard dont une classe permettant de reprendre des
        donnees de pe basees sur des caracteres semi-graphiques -->

<!ENTITY % qcm "INTRO?, CONSIGNE?, ITEM+">
<!ENTITY % simple-int "INTRO?, DATA">
<!ENTITY % situation "INTRO?, ENONCE, CONSIGNE?, MESSAGE?, FORM-REP?,
CONTROLE?, ACTIVITE?">
<!ENTITY % exercice "INTRO?, CONSIGNE?, QUESTION+, MESSAGE?, FORM-REP?">
<!ENTITY % fiche "INTRO?, (%elem.comp;)*">
<!ENTITY % analyse "RESOLUTION, SOLUTIONS">

<!ENTITY % content.content "(%qcm;) | (%simple-int;) | (%situation;) |
                (%exercice;) | (%fiche;) | (%analyse;)">
<!ENTITY % type.doc "qcm | simple-int | situation | exercice |
                fiche | analyse">

<!-- reprend les attributs du doc, utile pour les segments. A verifier
        la correspondance -->

<!-- %qcm; ne contient pas le container "QUESTION" il y une autre maniere
de faire des qcm: QUESTION - INTERACTION - ITEM+ -->

<!ELEMENT CONTENT (%content.content; | SEGMENT*)>
<!ATTLIST CONTENT TYPE (%type.doc;) #IMPLIED>

<!ELEMENT SEGMENT (%content.content;)>
<!ATTLIST SEGMENT TYPE (%type.doc;) #IMPLIED>

<!-- autres elements: RESOLUTION, SOLUTIONS
                QUESTION, INTERACTION, ITEM,
                DATA, SEGMENT, INTERACTION -->

<!-- ***** -->

<!-- *** Images *** -->

<!ELEMENT IMG EMPTY>
<!ATTLIST IMG
        SRC CDATA #REQUIRED
        WIDTH CDATA #IMPLIED
        HEIGHT CDATA #IMPLIED>

<!-- pourrait se transformer en un EMBED ou se fusionner
        avec l'element BBOARD qui lui-meme pourrait s'inspirer

```

```

    de l'element OBJECT de HTML -->

<!-- ***** -->

<!-- *** Anchors *** -->

<!ELEMENT A (%elem.base;)*>
<!ATTLIST A
    IDREF IDREFS #IMPLIED
    HREF CDATA #REQUIRED>

<!-- a voir dans le cas sans IDREF a pouvoir distinguer
les differents liens, par exemple :

<A HREF="javascript:parent.simpleReponse('',' ',4)">
<IMG SRC="images/pava4.gif" WIDTH="81" HEIGHT="81"/></A>

Cela demanderait d'ajouter un attribut TYPE et un
attribut PARAM.

HREF ne serait plus que le cas fourre tout en voie de
disparition.

A preferer NOTE ou CARTOUCHE ... A voir pour un element dont
le type par default serait "texte".

Remarques concernant les multi-liens

1)
<LNK IDREF="lien1" ...>
<LNK IDREF="lien2" ...>
...
<A IDREF="lien1 lien2">

2) A eviter ?
<LNK IDREF="lien" ...>
<LNK IDREF="lien" ...>
...
<A IDREF="lien">

-->

<!-- ***** -->

<!-- *** Formules *** -->

<!ELEMENT FORMULE ANY>
<!ATTLIST FORMULE
    FORMAT (tex | mathml) "tex"
    CODE CDATA #IMPLIED>
<!-- ANY pour mathml -->
<!-- voir le dispositif wikipedia pour tex -->

<!-- ***** -->

<!-- *** Table *** -->

<!ELEMENT BBOARD (CAPTION?, LABEL*)>
<!ATTLIST BBOARD
    WIDTH CDATA #REQUIRED
    HEIGHT CDATA #REQUIRED
    PRGM CDATA #REQUIRED>
<!-- A etendre pour en faire differents panels

```


pourrait contenir du SVG ou selon le cas devenir un EMBED
voir aussi le commentaire de l'element IMG -->

```
<!ELEMENT CAPTION (#PCDATA)*>

<!ELEMENT LABEL (#PCDATA)*>
<!ATTLIST LABEL
    POS CDATA #REQUIRED>

<!ELEMENT BLOC (CASE+ | LIGNE+)>
<!ELEMENT LIGNE (CASE)+>
<!ATTLIST BLOC
    DISPO (juxta | super | table) "juxta"
    COLS CDATA #IMPLIED
    BORD CDATA #IMPLIED>
<!ELEMENT CASE (%elem.plus;)*>
<!-- voir itinerai/nu_pb4.xml pour un exemple
    ou probleme1/cpi-exe-int03.xml
    voir qrep2-2.xsl pour la r√alisation des r√gles xsl.
    Question: est-il bien vrai, qu'en standard, qu'au cas ou plusieurs
    r√gles
    s'appliquent seule la premi√re est utilis√e -->

<!-- ***** -->

<!-- *** Phrases, notes *** -->

<!ELEMENT PHRASE (%elem.mini;)*>
<!ELEMENT NOTE (%elem.base;)*>
<!ATTLIST NOTE
    IDREF CDATA #IMPLIED
    NAME CDATA #IMPLIED
    TEXT CDATA #IMPLIED
    TYPE CDATA #IMPLIED>

<!-- <NOTE NAME=ancre IDREF></NOTE> -->
<!-- pas pris <NOTE NAME=ancre TYPE>la note</NOTE> -->
<!-- <NOTE NAME=ancre TYPE TEXT=lanote></NOTE> -->

<!-- <NOTE IDREF>zone notee</NOTE> -->
<!-- <NOTE TYPE TEXT>zone notee</NOTE> -->
<!-- le type popup, note, texte est donne dans le link
    dans le cas sans IDREF on utilise l'attribut TYPE
    par default le type est "note". A creer ev. un element
    de lien dont le type par default est "texte". -->

<!-- iden pour CARTOUCHE -->
<!-- <CARTOUCHE NAME=ancre IDREF></CARTOUCHE> -->
<!-- <CARTOUCHE NAME=ancre TEXT=lanote></CARTOUCHE> -->

<!-- <CARTOUCHE IDREF>zone notee</CARTOUCHE> -->
<!-- <CARTOUCHE TYPE TEXT>zone notee</CARTOUCHE> -->

<!-- ***** -->

<!-- *** Valeur d'une variable locale a la salle *** -->

<!ELEMENT VAL-VAR EMPTY>
<!ATTLIST VAL-VAR
    NAME CDATA #REQUIRED
    TYPE CDATA #IMPLIED
    FORMAT CDATA #IMPLIED>
```

```

<!-- permettre aussi variable globale
      et l'accès à une valeur retournée par une fonction
      via le type
      TYPE fct - fonction
      TYPE glo - variable globale
      défaut variable locale

      FORMAT img
      -->

<!-- ***** -->

<!-- *** Requete sur db *** -->

<!ELEMENT VAL-CHAMP EMPTY>
<!ATTLIST VAL-CHAMP
      NAME          CDATA #REQUIRED
      DATASOURCE    CDATA #REQUIRED
      SELECT        CDATA #REQUIRED
      FROM          CDATA #REQUIRED
      WHERE         CDATA #IMPLIED
      TYPE          CDATA #IMPLIED>

<!-- A prévoir aussi éventuellement via un link
      à permettre aussi sous la forme d'un élément ou la
      requête est le contenu (à la manière d'un script ou
      structure en sous éléments -->

<!-- ***** -->

<!-- *** Blocs *** -->

<!ELEMENT INTRO          (%elem.plus;)*>
<!ELEMENT PROPOSITION   (%elem.base;)*>
<!ELEMENT NOTION        (%elem.mini;)*>
<!ATTLIST NOTION
      NOM CDATA #IMPLIED>

<!-- les notions données par un nom sont des éléments vides -->
<!ELEMENT EXPRESSION   (%elem.base;)*>
<!ELEMENT EXEMPLE     (%elem.base;)*>
<!ELEMENT DEFINITION  (NOTION, EXPRESSION, EXEMPLE*)>
<!ELEMENT CITATION    (%elem.base;)*>
<!ELEMENT CONSIGNE    (%elem.plus;)*>
<!ELEMENT ENONCE      (%elem.plus;)*>
<!ELEMENT MESSAGE     (%elem.plus;)*>
<!ELEMENT CONTROLE ANY>
<!ELEMENT ACTIVITE ANY>

<!-- à définir encore la notion de contrôle et d'activité : à développer
      la feuille de style activite.xml -->

<!-- THEOREME ? -->

<!-- *** Formulaire *** -->

<!ELEMENT FORM-REP (%elem.plus; | INPUT)*>
<!ATTLIST FORM-REP
      TYPE          CDATA #IMPLIED
      SIZE          CDATA #IMPLIED
      SCHEME        CDATA #IMPLIED
      ALIGN         CDATA #IMPLIED
      VALUE         CDATA #IMPLIED>

<!-- type / scheme actuel:

```

```

entier | decimal / textuel (usage du modele) ;
entier | decimal / apply (usage de verifReponse() );
bouton / apply

    ALIGN="center"

    VALUE utiliser par grep-mult (entre autres) pour donner
    la r√@ponse attendue.
-->

<!ELEMENT INPUT EMPTY>
<!ATTLIST INPUT
    TYPE CDATA #REQUIRED
    NAME CDATA #REQUIRED
    SIZE CDATA #IMPLIED
    VALUE CDATA #IMPLIED>
<!-- prévu pour le cas TYPE="bouton" pour pouvoir mettre des
champs additionnels. A voir s'il ne faudrait pas passer
en parametre le formulaire a verifReponse()
-->

<!-- ***** -->

<!ELEMENT RESOLUTION (INTRO,ETAPE*)>
<!ELEMENT ETAPE (%elem.plus;)*>
<!ATTLIST ETAPE
    TITRE CDATA #IMPLIED>
<!-- a voir lien avec syst. expert -->

<!ELEMENT SOLUTIONS (INTRO, UNE-SOL*)>
<!ELEMENT UNE-SOL (%elem.plus;)*>
<!ATTLIST UNE-SOL
    TITRE CDATA #IMPLIED
    EVAL CDATA #IMPLIED
    SCHEME CDATA #IMPLIED>

<!ELEMENT QUESTION (ENONCE?, CONSIGNE?, INTERACTION)>
<!ATTLIST QUESTION
    NAME CDATA #IMPLIED
    DISPOSITION CDATA #IMPLIED>

<!ELEMENT INTERACTION (INTRO?, CONSIGNE?, ATTLIST?, CTXTLIST?, ITEM*)>
<!ATTLIST INTERACTION
    TYPE CDATA #REQUIRED
    WIDTH CDATA #IMPLIED
    HEIGHT CDATA #IMPLIED>

<!-- anciennement DATA-COMP a la place de CTX et ATT -->
<!-- alternative a item, ou alors elargir la notion -->
<!-- attribut ou contexte general pour signaler si la liste des items est
servie
d'un seul coup ou un a un -->

<!ELEMENT DATA (DATA-TYPE?, DATA-COMPL*)>
<!ELEMENT DATA-TYPE EMPTY>
<!ELEMENT DATA-COMPL EMPTY>
<!ATTLIST DATA-COMPL
    NAME CDATA #REQUIRED
    VALUE CDATA #IMPLIED>

<!-- NAME : consigne, remp, type, mode, rows, cols, align, sep
content -->

```

```

<!-- a mettre des infos type: lisibilite (difficulte),
heritees ou redefinies dans les items -->

<!ELEMENT ITEM (CONSIGNE?, ENONCE?, (FORM-REP | CHOICE+))> <!-- qcm -->
<!ATTLIST ITEM
  NAME CDATA #IMPLIED>

<!ELEMENT CHOICE (#PCDATA)>
<!ATTLIST CHOICE
  STATE (juste | faux) "faux">

<!-- a faire ev. des nouveaux elements
  SEGMENT TYPE="analyse" devient ANALYSE -->
<!-- CHOICE a ete ameliore, cf rev-bjn -->
<!-- voir comment situer les images cliquables -->

<!-- ***** FIN DE LA DTD ***** -->

```

```

<!-- etude des feuilles de style existantes et des modifications
a apporter selon cette nouvelle version

```

qrep.xml / qrep2.xml (exercice)

```

PEXP
  HEAD
    TITLE
    ATTLIST
    VARLIST
    SCRIPT
    LINKS
  CONTENT
    DIV
    INTRO
      VAL-VAR
    FORM-REP
      VAL-VAR
    INPUT

```

(a introduite DIV dans INTRO)

qcm2.xml (qcm)

```

PEXP
  HEAD
    TITLE
    ATTLIST
    CTXLIST
    LINKS
  CONTENT
    INTRO
    ITEM*
      TEXT
      CHOICE*

```

(remplacer TEXT par ENONCE)

applet.xml / applet2.xml (simple-int)

```

PEXP
  HEAD
    TITLE

```

```

                ATTLLIST
                CTXLLIST (contient nom de l'applet)
                LINKS
CONTENT
                INTRO
                DATA
                        DATA-COMPL*

```

(cas eventuellement a eviter)

qtml.xml / vtml.xml

```

PEXP
    HEAD
        TITLE
        ATTLLIST
        VARLLIST
        LINKS
    BODY

```

lacune.xml (exercice)

```

PEXP
    HEAD
        TITLE
        ATTLLIST
        CTXLLIST (contient seul. l'indication de l'interaction
lacune
        LINKS
CONTENT
        INTRO
        QUESTION*
                DATA
                        DATA-COMPL*

```

(remplacer l'interieur de question par: INTERACTION avec
a l'interieur ATTLLIST et/ou CTXTLLIST)

fiche.xml (fiche)

```

PEXP
    HEAD
        TITLE
        ATTLLIST
        LINKS
CONTENT
        INTRO
        DEFINITION*
                TEXT
                BBOARD*

```

(revoir le contenu de definition)

fiche2.xml (exercice)

```

PEXP
    HEAD
        ATTLLIST
CONTENT
        INTRO
        QUESTION
                ENONCE
                INTERACTION
                        DATA-COMPL

```

(remplacer DATA-COMPL par ATTLIST et/ou CTXTLIST)

-->