**TASC - ALPHALINC - PENDÊNCIAS TÉCNICAS (NÃO FUNCIONAIS)**

|  |  |
| --- | --- |
| **Componente** | **Pendência** |
| ORM | Revisão e testes dos métodos já implementados  - Revisada geral nos método previstos no mEntity (confirmar métodos pendentes de implementação)  - Utilização da nova estrutura de lista de relacionamentos (mEntityList) |
| ORM | Tratamento de relacionamentos com referência para chave única (sem referência para PK)  - Validar tratamento com a nova estrutura de lista de relacionamentos (mEntityList) |
| ORM | Implementação dos métodos pendentes para uso futuro  - Relacionar os métodos pendentes |
| ResultSet | Tratamento para mapeamento e conversão de código (recurso multi-banco focado principalmente nas funções $$)  - Construir uma classe no netmanager (ou no alphalinc) que converte as funções $$ conforme determinados parâmetros de entrada (este método desta classe será acionado pelo ResultSet através de uma variável do contexto) |
| ResultSet | Adequação da COMView para recuperar os campos diretamente via SQL (eliminar necessidade de cache de dados via global)  - Shobby já está trabalhando nesta adequação |
| ResultSet | Tratamento de select distinct com CLOB => utilizar um row\_number() over (partition by <columns>)  - Este problema será minimizado com a adequação da COMView  - Caso o problema ocorra em algum comando SQL deve-se ajustar para eliminar o distinct |
| ResultSet | Tratamento de translate com CLOB => verificar viabilidade de utilização de uma função java no Oracle  - Definir tratamento para restringir dados (where) e realizar o translate com campos CLOB (?) |
| ResultSet | Tratamento de order by com CLOB => truncar os valores com substring  - Truncar valor CLOB e ordenar os dados até 4000 caracteres |
| ResultSet | Tratamento para conversão de data (condição where e outras situações ...)  - Criar um facilitador para customizações de queries SQL com utilização de parâmetros do tipo date (criar um método que permite a identificação do tipo de dado do parâmetro conforme sua posição) |
| ORM-JDBC | Tratamento de update com CLOB => utilizar a função concat e a conversão dos strings através to to\_clob()  \*\* RESOLVIDO |
| ORM-JDBC | Pesquisa de índices para popular globais "s" via banco de dados relacional  - Nos comandos $data, $order e get de globais popular o cache de índices de classes relacionais através de uma query no banco relacional  - Determinar identificação do índice pelo segundo nível do subscrito (por exemplo ^INARTs(0,5) => utiliza o índice 5 da INART)  - Recuperar estrutura de dados do índice na WWWCLASSMAPPING (sugestão de criar um cache da estrutura em uma global do tipo 92 WWWCLASSMAPINDEX) |
| NetManager | Geração de índices em globais "s" somente para dados em globais (90 ou 91)  \*\* RESOLVIDO |
| NetManager | Utilitário para recriação de índices em globais "s" (\*\*Shobby)  \*\* RESOLVIDO |
| Conversor | Revisão de tratamento de referência de objetos encadeados (métodos e propriedades) (\*\*Alexandre)  \*\* RESOLVIDO |
| Conversor | Implementar controle de transação via JTA (\*\*Alexandre)  \*\* RESOLVIDO |
| Framework | Ajustar utilização do hashmap para controle de cache de globais (atualmente não utilizada o hashmap de hashmap)  - Adequar o controle de indicação de globais no cache => QueryCache (utilizar mesma estratégia de mapa de mapa empregada nos dados do cache) |
| Framework | Armazenar a referência direta ao Node nas variáveis locais sem subscrito (corrigir problema de redeclaração da variável utilizada em passagem de parâmetro por referência)  - Incluir no mVar referência ao Node somente de variáveis locais sem níveis de subscrito (o Node seria populado no primeiro comando de set da variável)  - No newVar teria forçar o Node ficar nulo (gerar um novo node no próximo comando set)  - O comando restore tem que marcar os Nodes removidos (de forma que o mVar identifique que o Node foi removido e deve recuperar o Node corrente)  - Na passagem de parâmetro por referência deve forçar a inicialização do Node (de forma que o parâmetro aponte para a variável de origem) |
| Framework | Tratamento para otimização de laços com referência de um nível de outra global (inteligência para gerar o cache automático)  - Analisar opções para otimizar a recuperação de dados de globais em laços com $order baseados em outra global (uma opção seria criar um comando diferenciado para estas situações em que o get recupere um conjunto de dados via like) |
| Framework | Revisar implementação do $data (limitar verificação de dados de uma tabela relacional)  - Limitar dados para popular o cache no comando $data com tabela relacional (100 registros ???)  - Avaliar impacto e estratégia para o comando $order (???) |
| Framework | Limpeza automática do cache de globais (determinar regras e controles para limpeza)  - Analisar estratégias para limpeza do cache  - Estratégia inicial:  \* Limpar diariamente os dados de cache de tabelas relacionais  \* Limpar diariamente os dados de globais de histórico  \* Limpar diariamente os dados de globais de sessões encerradas |
| Framework | Implementar controle de transação via JTA (incluir tratamento nos códigos já convertidos)  - Falta revisar os códigos já convertidos que possuem controle de transação (tstart, trollback, tcommit, SQL(start transaction), SQL(rollback), SQL(commit)) => incluir chamada dos comandos do framework |
| Report | Implementação de um componente personalizado (resultset) para recuperação dos dados  - Analisar viabilidade de criar um componente customizados no Jasper Reports que acesse o ResultSet do framework (alternativa via JDBC)  - Analisar a utilização de um componente com acesso aos dados via JPA EcliseLink para o futuro |
| Library | Implementação das funções de composição e decomposição de Status (JIRA)  - Liberar até sexta-feira (Alexandre) |
| NetManager | Geração dos comandos DDL para criação/alteração/remoção de tabelas no banco relacional (sincronizar DDL com o mapeamento das classes e gerar histórico de comandos gerados, bem como, reversão dos comandos submetidos)  - Analisar processo atual de geração da DDL para migração do banco (POC1)  - Criar um novo componente para geração da DDL e atualização do mapeamento (estas informações devem estar sincronizadas)  - Os comandos DDL submetidos com sucesso devem ser armazenados para serem aplicados futuramente na atualização de versão do sistema (deployment)  - Os comandos DDL com falha devem ser apresentados para o usuário de forma que sejam feitas as correções do dicionário de dados no netmanager |
| Deployment | Definir e implementar os controles necessários para liberação de versão (fontes, meta-dados, configurações, DDL, etc.) |
| Framework | Substituição do ORM-JDBC pelo novo ORM (tornar o sistema independente do banco de dados com o uso do JPA) |
| NetManager | Separar as classes do netmanager das classes do sistema Alphalinc (criar novo projeto Alphalinc)  - Revisar classes que pertencem ao netmanager (a princípio as classes WWW e COM são netmanager e as demais são alphalinc) |