DOM4J a SAX

Plantlore hojně využívá technologie XML. V XML formátu jsou uloženy konfigurační soubory a do tohoto formátu musí umět Plantlore provádět export záznamů z databáze a stejně tak musí umět z XML načítat data pro následný import záznamů.

V případě práce s XML jsme nechtěli opakovat již provedenou práci a psát si vlastní parsery XML, použili jsme známé a prověřené knihovny SAX a DOM4J.

DOM4J

DOM4J umožňuje snadné vytvoření reprezentace XML stromu ze souboru. Mezi jeho slabiny patří zejména vysoké paměťové nároky, proto jej bylo lze použít pouze pro případy malých konfiguračních souborů nebo rozdílových seznamů pro import dat přímo do tabulek, které slouží k jejich opravě či doplnění.

Ukázalo se, že v obou případech (konfig. soubory i rozdílové seznamy) je použití DOM4J ospravedlnitelné - konfig. soubory jsou velmi malé a rozdílové seznamy mohou obsahovat řádově tisíce položek, aniž by bylo zapotřebí extremních paměťových nároků (nejobsáhlejší záznamy tabulek TPlants je možné zpracovávat v řádech tisíců, přičemž paměťové nároky jsou několik desítek MB, např. soubor 5tis rostlin potřebuje cca 20MB paměti, což při velikostech dnešních operačních pamětí nepředstavuje žádný problém). Pro představu uveďme, že v celé České republice je rozpoznáváno zhruba 6tis větších sídel a aplikace Plantlore pracuje se zhruba 5tis druhy rostlin. Je nutné si uvědomit, že zamýšlené použití pro import do tabulek je oprava některých záznamů nebo přidání nových záznamů - a v případě větších sídel nebo rostlin nehrozí nárůst více než o několik desítek záznamů ročně.

DOM4j byl dále použit pro konstrukci výstupu při exportu záznamů z databáze. Zde bylo jeho použití poněkud upraveno, jelikož by nebylo možné postavit DOM strom v paměti pro několik tisíc nálezových záznamů, natož pak pro avizované desetitisíce (zadavatelé nashromáždili přibližně 50tis nálezových dat). DOM4J byl použit pro konstrukci XML stromu vždy pouze pro jeden záznam a pak okamžitě zapsán na výstup; zbývající XML tagy byly vygenerovány ručně. Tímto bylo efektivně využito výhod, které DOM4J nabízí (především snadné generování stromu a jeho zápis do souboru), aniž by to vedlo na zvýšené paměťové nároky. Export vyžaduje paměť v řádu desítek kB během celého procesu.

SAX

Pro import nálezových dat nebylo již DOM4J možno vůbec použít z důvodu velikosti vstupních dat. Proto byl použit (pro programátora trochu méně příjemný) způsob parsování souboru pomocí SAX, který pouze reportuje nastalé události (nalezení tagů, text mezi tagy), a je na programátorovi, aby z daných událostí sestavil adekvátní strom - v našem případě tedy rekonstruoval čtený záznam.

Užití SAXu nás zbavilo problémů spojených s parsováním souboru, ale znemožnilo přímou aplikaci návrhového vzoru Builder, protože SAX pracuje na principu zpětného volání (callback); architektura importu nálezových dat je postavena odlišně.

Závěr

Obě technologie našly své uplatnění a zjednodušily zpracování XML souborů, třebaže u DOM4J bylo zapotřebí pečlivě monitorovat spotřebovanou paměť a u SAXu poupravit architekturu aplikace. Při jejich použití se žádné problémy nevyskytly.