ЛИНГВИСТИЧЕСКОЕ ОБЕСПЕЧЕНИЕ РАСПРЕДЕЛЕННЫХ АВТОМАТИЗИРОВАННЫХ СИСТЕМ УПРАВЛЕНИЯ НА БАЗЕ UNIFIED MODELING LANGUAGE.

Лосев В.В.
ГОУ ВПО СибГТУ, Красноярск, Россия

Среди приоритетных задач обеспечения функционирования GRID-системы, как архитектурной платформы для распределенных автоматизированных систем управления (АСУ) выделим задачу обмена данными, посредством основной памяти узлов, при взаимодействии прикладных программ с системами управления базами данных (СУБД). С последующим представлением данного метода с помощью информационной модели UML, как лингвистического обеспечения распределенной АСУ.

Рассматривая взаимодействие систем класса SCADA (Supervisory Control And Data Acquisition) с традиционными СУБД, ориентированными на дисковую память, выделим задачу загрузки объекта сущности с целью применения вычислительных процедур со стороны SCADA-системы.

Описание данной модели представлено в виде UML – диаграммы последовательностей Рис.1. Инициатором взаимодействия систем выступает объект *Arc.Trend*, представляющий архивный тренд в SCADA-системе, который, в свою очередь, осуществляет запрос объекта *N.Val.Param*, представляющий n-мерный массив значений параметра технологического процесса (ТП) из таблицы *Val.Param*, с целью последующего представления (визуализации). Объекту *Int.Mng*, представляющему менеджер интерфейса SCADA-системы известно об отсутствии в памяти объекта ***N.Val.Param,*** рассмотрение данного механизма предполагает UML - диаграмма деятельности. Данное известие влечет за собой попытку обращения к БД. Для реализации поиска объект *Arc.Trend* посылает атрибут *tempr.Val* (значение температуры), как условие поиска, в качестве аргумента операции *getVal.Param ()*.

Следующим действием является создание объектом *Int.Mng* строки SQL-запроса и передачи его объекту ***Int.Drv,*** представляющему драйвер интерфейса в сообщении *query ()*. Объект *Int.Drv* реализует механизм обращения к БД, который предполагает операции по синтаксическому анализу, проверке авторизации и компиляции. Результатом данного обращения является получение данных из таблицы *Val.Param*, с последующим их возвращением объекту *Int.Mng*.

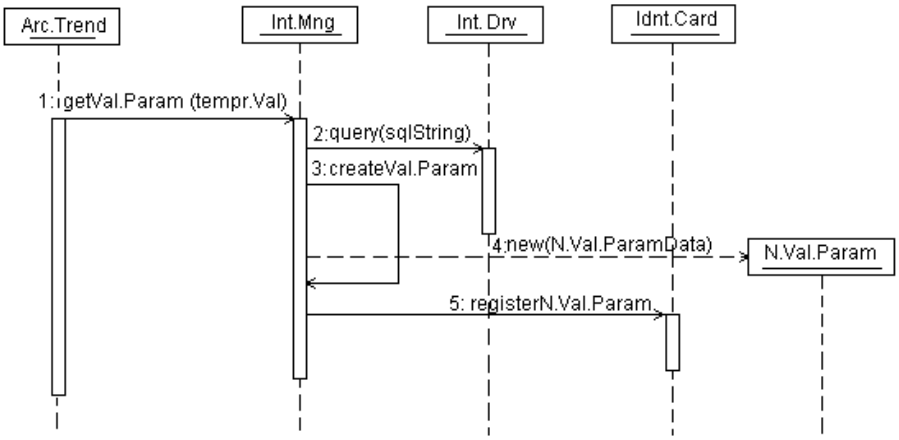

Рис. 1.- UML – диаграмма последовательностей загрузки объекта.

Процесс создания объекта *N.Val.Param* реализуется в процессе инициализации метода *createVal.Param ()*, объектом *Int.Mng*. Данный метод отвечает за создание нового объекта *N.Val.Param* с помощью сообщения *new ()*. Завершением процесса загрузки нового объекта *N.Val.Param* является его регистрация объектом *Idnt.Card,* в результате присвоения идентификатора «чистый», по запросу объекта *Int.Mng* соответствующим флагом.

Анализируя полученную диаграмму следует отметить, что вызов персистентных объектов, имеет место при взаимодействии с СУБД, ориентированной на дисковую память и влечет временные задержки при совершении ряда действий.

Успешный результат поиска объекта сущности в памяти предопределяет новую задачу. Необходимо удостовериться, что объект является «чистым», т.е. содержит текущие данные хранящиеся в БД.

Запись значений параметров ТП, следует осуществлять короткими, легковесными транзакциями, характерными для OLTP (On-Line Transaction Processing). Продолжительность выполнения транзакции зависит от частоты опроса, например, устройств полевых сетей. Данное значение будет соответствовать длительности транзакции, и как следствие, «чистоте» данных и их доступности. Например, при запросе чтения объектом *Arc.Trend* данных незавершенной транзакции, результатом запроса будут «грязные» данные, либо блокировка на доступ к данным, до завершения транзакции с последующим выталкиванием данных на дисковую память, откуда и будет произведена загрузка объекта.

Рассматривая GRID-систему как архитектурную платформу для распределенных АСУ, с возможностью размещения в основной памяти узлов OLTP-ориентированной СУБД позволит исключить временные задержки при взаимодействии с дисковой памятью.

В данном случае станет применима однопотоковая (сериализуемая) модель выполнения транзакций.

Пересмотр интерфейса взаимодействия позволит исключить последовательность действий направленных на синтаксический анализ, проверку ссылок на объекты, оптимизацию и компиляцию. В альтернативу SQL-запросам, со стороны SCADA-систем, посредством интерфейсов, следует рассматривать хранимые процедуры, например, предопределенные транзакции. При соблюдении условия формирования коротких (предопределенных) транзакция, возрастет доступность данных (их «чистота»), что позволит приблизить скорость реализации модели загрузки объектов к реальному времени.