МЕТОДЫ ПОВЫШЕНИЯ НАДЕЖНОСТИ СИСТЕМЫ УПРАВЛЕНИЯ ИНТЕЛЛЕКТУАЛЬНЫМ ГОРОДОМ

Гладий Л.В., Халава Саид Фауаз
Харьковский национальный университет радиоэлектроники
61166, Харьков, пр. Ленина,14, каф. Телекоммуникационных систем,
тел.(057) 702-55-92), E-mail: tcs@kture.kharkov.ua; факс: (057) 702-13-20

The concept an intelligent city is a new stage for development of an infrastructure of a city with considerable improvement of living conditions of its inhabitants. Service-oriented architecture- is one of the most perspective technologies of construction of the distributed global network. Its basic advantages are loosely coupled of applications, independence of the programming language and platform that is especially important for a heterogeneous network of a city. Reliability and ensuring availability of services is one of the most important characteristics of such network. In working it is offered to use a method dynamic replication for ensuring availability of services.

В настоящее время одним из наиболее перспективных направлений развития науки и техники является обеспечение комфортных и безопасных условий для жизни людей. Наиболее масштабным проектом, реализуемым в этом направлении, является проект «интеллектуальный город», в основе которого лежит идея создания единого информационного пространства, позволяющего осуществлять управление городом, обеспечивать безопасность жителей и объектов городского хозяйства, а также осуществлять мониторинг состояния главных городских объектов. Программа «интеллектуальный город» ориентирована на модернизацию и реконструкцию существующей инженерной сети, с целью создания единого глобального информационного пространства, в которое будут подключены службы различных городских объектов.

В качестве примера рассмотрим работу сети коммунальных служб города (рис.1).

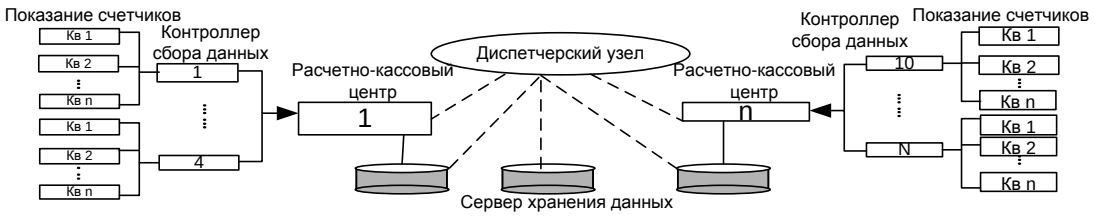

Рис. 1 – Структурная схема сети коммунальных служб города

В каждой квартире или доме устанавливается счетчик на холодную и горячую воду. Один раз в сутки показания со счетчика снимаются и передаются на контроллер сбора данных, где информация обрабатывается и передается в ближайший расчетно-кассовый центр. В расчетно-кассовом центре производится оценка показаний счетчиков и определяется оплата за предоставляемые услуги. Затем данная информация отправляется на личный счет клиента, который хранится на одном из серверов сети. Клиент посредством сети интернет может зайти на свой личный счет и проверить задолженность по оплате и сверить показания счетчиков. Также с помощью терминалов, банков и сети интернет он может произвести оплату предоставляемых ему услуг. Центральным узлом для всей системы является диспетчерский центр, который имеет доступ к базе данных клиентов и к расчетно-кассовым центрам, что упрощает процесс внесения изменений в систему и контроль ее состояния. Такая структура системы управления является более надежной сравнительно с централизированной, так как выход из строя любого узла сети не повлияет на работу всей системы.

Важным вопросом, решаемым при создании такой городской сети, является выбор ее архитектуры и метода управления. В связи с тем, что основными требованиями, выдвигаемыми к сети, является гибкость системы, отказоустойчивость, масштабируемость и надежность, то наиболее подходящим было бы использовать распределенную система

управления ресурсами сети. В качестве архитектуры предлагается использовать сервис-ориентированную архитектуру (SOA), основными преимуществами которой являются слабая связность приложений, независимость от языка и платформы, что является особенно важным для разнородной городской сети.

В самом общем виде сервис-ориентированная архитектура предполагает наличие трех участников: поставщика сервисов, потребителя сервисов и реестра сервисов [1]. Их взаимодействие представлено достаточно простой схемой: поставщик регистрирует сервисы в реестре, а потребитель обращается к реестру с целью получения адреса необходимого ему сервиса. После получения адреса потребитель может отправлять запросы к необходимому ему сервису напрямую. В том случае, если в базе данных локального реестра сервиса нет запрашиваемого ресурса, запрос пересылается удаленному реестру. Если и там сервис не будет обнаружен, потребителю будет отправлено сообщение об отсутствии данного ресурса в сети. Каждый реестр имеет свою область обслуживания, границы которой определяются зарегистрированными в нем сервисами, другими словами, при регистрации услуг предприятия или личного счета жителя города в реестре указывается, что данный реестр сервисов является для него локальным и ни в каком другом реестре он до этого не был и не будет зарегистрирован.

Использование SOA-архитектуры является хорошим решением для организации единой городской сети, однако сервис-ориентированная архитектура имеет один существенный недостаток – отсутствие методов обеспечения доступности сервисов, что может значительно повлиять на характеристики такой сети. Пока количество предоставляемых услуг будет не большое, вопросы доступности не будут иметь особого значения, но в дальнейшем при росте количества умных домов, возникновении таких услуг, как проверка датчиков состояния квартиры в реальном времени, возникновении новых услуг предоставляемых предприятиями города, ситуация может измениться. В связи с этим в данной работе будут рассмотрены существующие методы повышения надежности и, в частности, доступности, с целью адаптации их к SOA

Одним из таких методов является репликация, которая используется в архитектуре CORBA. Однако, как показывает практика, повсеместное использование репликации из-за необходимости обеспечения непротиворечивости данных может привести к перегрузкам сети и ухудшению ее характеристик. Поэтому применение данного метода необходимо рассматривать для каждого определенного случая отдельно. Примером такого случая может быть ситуация, когда к определенной услуге предприятия города за небольшой промежуток времени возрастает количество запросов. С целью уменьшения передаваемого трафика по сети, а так же повышения доступности сервиса разумным решением будет реплицировать его на локальный сервер вблизи запрашиваемых его пользователей. В качестве метода репликации предлагается использовать метод, основанный на динамической репликации, описание которой представлено в [2], при этом модифицировать его для применения в SOA.

В соответствии с алгоритмом динамической репликации (рис. 2) локальный реестр сервисов должен по истечению определенного промежутка времени проверять счетчики запросов к сервисам. Если счетчик обращений реестра $cnt(R,F_i)$ превысит порог репликации $rep(F)$, то реестр придет к решению о необходимости проведения репликации данного сервиса. Прежде чем проводить данную операцию, реестр просматривает свою базу данных на наличие сервера, который имеет ресурсы, необходимые для размещения данной реплики. Если такой сервер имеется, реестр посылает запрос к сервису F на создание копии, при этом в запросе он указывает месторасположение для будущей копии. После проведения операции репликации, реестру отправляется ответ, в котором указывается, была ли успешно выполнена данная процедура и адрес для регистрации реплики. Данная информация также будет отослана на удаленный реестр сервисов для создания отметки копии и указания ее адреса, в том случае, если реплицированный сервис не являлся локальным. Если в базе данных реестра доступного сервера не обнаружено, то репликация отменяется, и реестр переходит к проверке счетчика следующего сервиса. По окончанию

процедуры проверки все счетчики обнуляются. Если общее число обращений к реплике упадет ниже порога удаления, то с целью освобождения места на сервере запускается процесс ее удаления. Если счетчик обращений реестра *cnt(R,F$_i$)* меньше порога репликации *rep(F)* и больше порога удаления *del(F)* счетчик обнуляется и реестр переходит к обработке счетчика следующего сервиса.

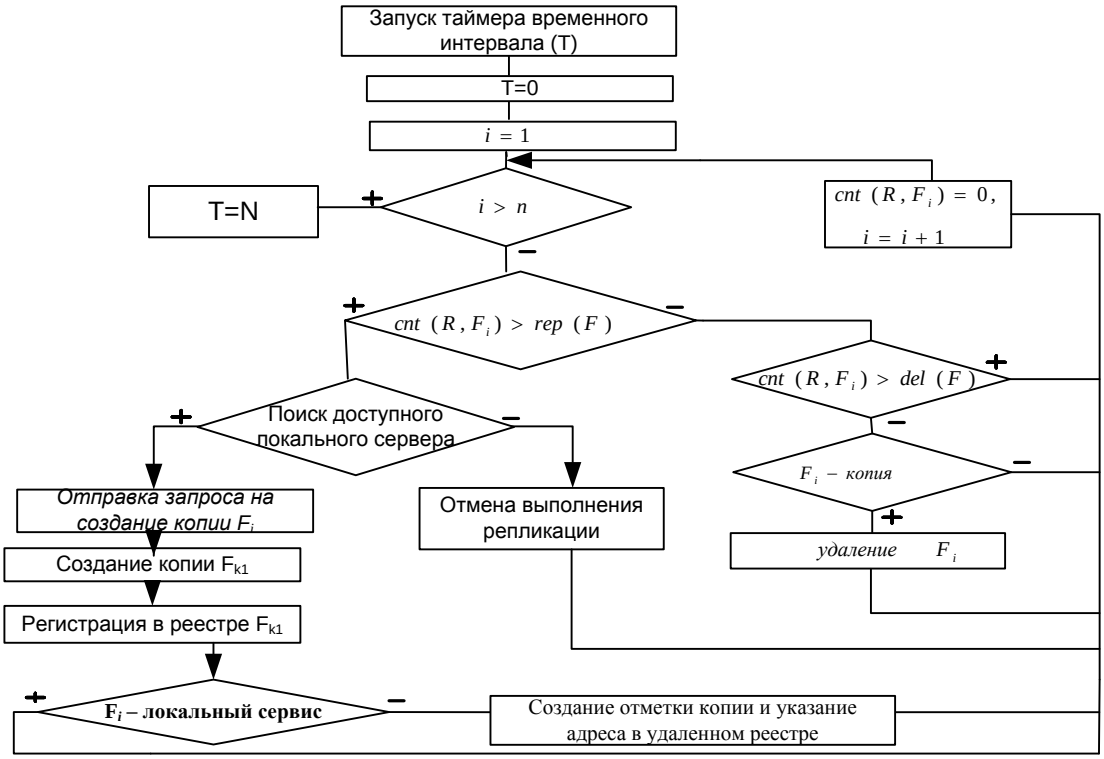

Рис. 2 – Алгоритм динамической репликации для SOA систем

Создание дополнительной реплики позволяет повысить доступность запрашиваемого сервиса, путем перераспределения запросов между репликами. Если во время прихода запроса сервис F будет занят или находится в нерабочем состоянии, запрос автоматически будет перенаправлен к реплике F_{k1}, при этом данное действие будет прозрачным для потребителя. Если сервис является удаленным, то, при приходе нового запроса на реестр, пользователю будет возвращен адрес сервиса F_{k1}. Основное преимущество использования репликации для удаленного сервиса состоит в том, что доступ к сервисам может выполняться локально, без поглощения сетевого трафика и задержек. Однако репликация сервиса не всегда может быть выполнима, это касается тех случаев, когда локальный сервер уже сильно перегружен или в нем отсутствует место на диске. В этом случае ищется альтернативный сервер или процесс репликации отменяется до окончания следующего временного интервала.

Как было указано ранее, важным вопросом при внедрении репликации является требование непротиворечивости данных. Согласно алгоритму динамической репликации, всегда существует первичная копия сервиса, которая даже при низком уровне обращений не удаляется, и которую имеет право изменить только поставщик. Поэтому для обеспечения непротиворечивости данных в SOA целесообразно использовать протоколы на базе первичной копии. Одним из наиболее распространенных протоколов такого вида является протокол первичного архивирования, рассмотренный в [2]. В данной работе предлагается модифицировать данный протокол с целью применения к сервис-ориентированным системам. Принцип работы данного протокола представлен на рисунке 4. Здесь используются следующие обозначения: 1) запрос на внесение изменений в сервис; 2) запрос к

реестру на наличие реплик и выдачу их адресов; 3) ответ реестра; 4) сигнал об обновлении резервных копий; 5) подтверждение обновления; 6) подтверждение внесения изменений.

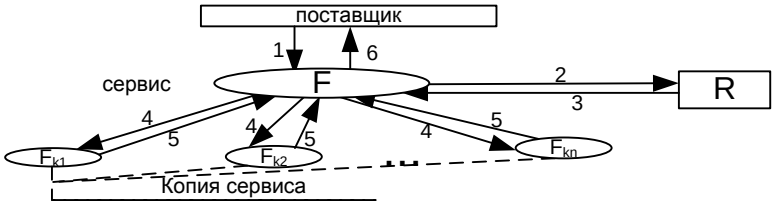

Рис. 4 – Протокол первичного архивирования

Поставщик для проведения операции изменения данных отправляет запрос к сервису, в который он хочет внести изменения. Данный сервис осуществляет обновления своих данных, после чего отправляет запрос к локальному реестру сервисов на наличие отметок копий и их адресов. Если таких копий не существует, процесс обновления завершается, и сервис вновь становится доступным. Если в ответе реестра будет указано наличие копии, то сервис отправит данные обновления всем его репликам. После прихода подтверждений об обновлении от всех реплик, сервис посылает поставщику подтверждение о внесении изменений, после чего данный процесс считается завершенным.

Процесс обновления происходит как одна атомарная операция или транзакция, что обеспечивает непротиворечивость всех копий сервиса. Недостатком данного протокола является то, что во время обновления потребитель, которых желает получить необходимый ему сервис, будет находиться в процессе ожидания. Однако, в том случае, если сервисы в SOA будут подвергаться модификации относительно редко, и право на их изменения будет иметь только поставщик, при небольшом количестве реплик время блокировки сервиса будет не значительным.

В случае роста количества реплик или же увеличения частоты их обновления можно будет использовать не блокирующий протокол обеспечения непротиворечивости. В отличие от предыдущего алгоритма, первичный сервис, сразу, после обновления своих данных, возвращает подтверждение поставщику и становится доступным пользователям. После этого он отправляет обновления всем своим репликам. Использование данного протокола практически не влияет на доступность сервисов при большом количестве реплик, но степень обеспечения непротиворечивости, а также защита от сбоев будет ниже. Предполагается, что поставщик, в зависимости от необходимой строгости обеспечения непротиворечивости данных, будет сам определять один из предложенных методов в соответствии с требованиями своего сервиса. Таким образом, при выборе оптимальной стратегии по применению репликации и модели обеспечения непротиворечивости, которая бы соответствовала особенностям систем, построенных на базе SOA, можно достигнуть высокой надежности и производительности сети.

Использование сервис-ориентированной архитектуры при построении единой городской сети позволит улучшить ее управляемость, быстродействие и надежность, а благодаря разработанному алгоритму репликации – и доступность сервисов.

Литература: 1. А.В. Богданов, Е.Н. Станкова, В.В. Мареев Сервис-ориентированная архитектура: новые возможности в свете развития GRID технологий / Всероссийский конкурсный отбор обзорно-аналитических статей по приоритетному направлению "Информационно-телекоммуникационные системы", 2008. – 32 с. 2. Э. Таненбаум, М. ванн Стеен. Распределенные системы. Принципы и парадигмы / Э. Таненбаум, М. ванн Стеен. — СПб.: Питер, 2003. — 877 с.