Web базирани модели за разпределена автоматизация

Николай Каканаков

Увод

През последните години развитието на компютърните мрежи направи достъпна глобалната компютърна мрежа - Интернет до все повече хора и организации. Наред с това развитието на информационните технологии позволи пренасянето им във всички сфери на обществото, като медицина, индустрия, аграрни науки, образование. Появиха се съвременни концепции за електронно обучение, електронен бизнес, електронно правителство и други. Тези концепции се пренасят и в разпределената автоматизация.

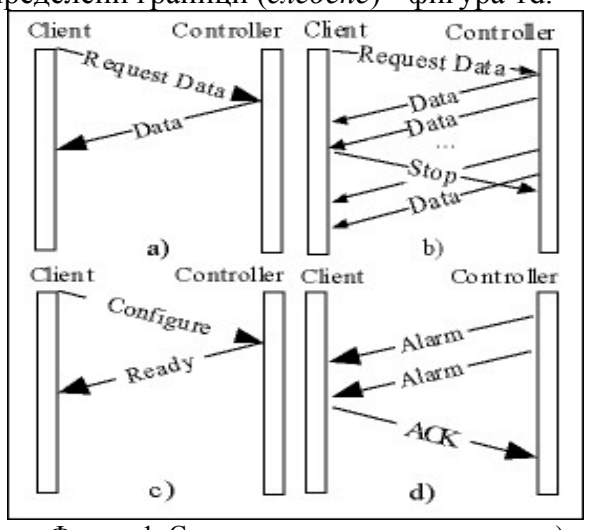
Наличието на вградени системи с интегриран комуникационен итерфейс и TCP/IP стек позволява в системите за разпределена автоматизация да се прилагат модели, доказали се в разпределените десктоп системи.

В настоящата статия са представени някои от най-популярните сървърни техологии, както и възможността за трансфер на тези технологии в системите за разпределена автоматизация.

Web базирани разпределени вградени системи

Такива системи включват два основни типа обекти – потребител и контролер. Тези обекти обменят данни по следните сценарии (фигура 1) [4]:

- *Следене/Контрол* – потребителят изисква от устройството резултата от определено измерване или изпълнението на определена команда – фигура 1a.
- *Диагностика* – потребителят изисква информация относно вътрешното състояние на устройството (апаратни и програмни грешки, захранване, статистика на работата и други) – фигура 1b.
- *Конфигуриране* – потребителят променя настройките на устройството. Така се променя хода на работа или се конфигурират аларми. Конфигурацията се запазва в набор от вътрешни променливи на устройството – фигура 1c.
- *Аларми* – при тях устройството е инициатор на обмена. То може да бъде конфигурирано да съобщава на потребителя за апаратна или програмна грешка в хода на работа (*диагностика*) и проверка дали следения от устройството параметър излиза от определени граници (*следене*) - фигура 1d.



Фигура 1: Сценарии за отдалечени услуги: a) диагностика, b) следене/контрол, c) конфигуриране, d) аларми.

За изпълнението на тези сценарии са необходими три вида комуникации [4]:

- *Заявка/Отговор* – този вид комуникация се използва за обмен на данни за диагностика или за конфигуриране.
- *Регистрация* – потребителя обявява, че иска да получава данни от определено измерване. След регистрацията ролите на клиент и сървър се сменят.
- *Спонтанна комуникация* – подобно е на заявка/отговор, но тук устройството е инициатор. Подходяща е за аларми.

За изпълнението на посочените сценарии се прилагат различни модели.

Моделите, прилагани в разпределената автоматизация, въпреки своите специфични изисквания, се базират на доказани в бизнес среда модели за обмен на данни. Специфичността на разпределените системи се проявява в две основни насоки – специфичните изисквания към актуалността на данните и ограничените ресурси на вградените системи.

Модели за разпределена автоматизация

1. Клиент/Сървър системи, базирани на уникален протокол.

Такива системи реализират отдалечено следене и контрол, като няма автоматична реакция. Вградената система е близко до контролирания обект и е свързана към Интернет през някаква физическа среда – безжично, телефонна линия, GPRS и други. Чрез специализирана програма на клиентската станция се предават командите и се получава следената информация. Такива системи са евтини за производство, но не се базират на стандартни протоколи, което създава трудности при по-сложни за следене групи от обекти. Подходящи са за диагностика и аларми, но не за мониторинг на параметри.

2. CGI (Common Gateway Interface) базирани разпределени вградени системи.

При тези системи изискванията към контролерите са по-големи. Необходим е мрежов интерфейс, TCP/IP стек и Web сървър. Контролерите, отговарящи на тези изисквания, станаха широкодостъпни през последните години [3].

Разпределените вградени системи, базирани на динамични HTML страници, могат да бъдат реализирани по два механизма [1, 3, 4].

Единият механизъм е значително по-прост и лесноизпълним, но е бавен и не се базира на стандартни решения. При изпълнението на работата си, вградената система периодично запазва HTML документ, съдържащ диагностичната информация и информацията за следения обект директно. Така през определено време се презаписват страниците с по-актуална информация. Недостатък е, че не може да се изиска страница, зависеща от клиентски параметри или да се предаде команда към контролера [4].

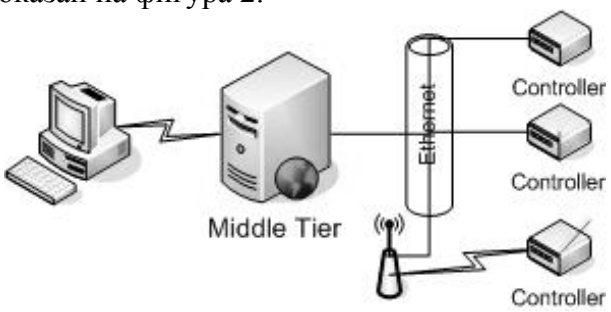
Вторият механизъм е стандартен CGI. Тук вече в отговор на клиентската http заявка не се връща статична HTML страница, а резултат от изпълнение на програма на контролера. Като всяка програма и CGI програмата може да получава параметри, но резултатът е един и се връща през стандартния изход, който в случая е пренасочен към http сървъра. По този начин можем да получаваме актуална информация за работата на контролера, както и да изпращаме команди. За аларми най-често се използва SMTP. В този механизъм не е предвидена комуникация от вида *регистрация* и *спонтанна комуникация*, тъй като контролерът не може да бъде инициатор [1, 3, 4].

3. Трислойна клиент/сървър архитектура.

Трислойната архитектура е развита основно при работата с бази данни. Системата с база от данни се състои логически от три части – потребителски интерфейс, програмна логика и база от данни. Това логическо разделение е довело до създаването на трислоен модел. На предния слой е потребителският интерфейс (front-end), на средния слой е бизнес логиката (middleware), а на третия слой е базата данни (back-end) [7, 8].

Тази архитектура може да се приложи успешно и в разпределената автоматизация. Потребителският интерфейс е същия, междинният слой спазва същите изисквания, но слоят за данни е представен от мрежа от контролери. Тези контролери следят параметри в реално време и се явяват източник на данни в трислойния модел [4, 5].

Трислоен модел с мрежа от контролери е показан на фигура 2.



Фигура 2: Трислоен модел с контролери

Така представената архитектура позволява разделение на презентационната част от автоматизационната част, както и записване на журнална информация от цялата мрежа от контролери на сървъра от междинния слой, който има повече ресурси от контролерите. Друго предимство на тази архитектура е сигурността – мрежата от контролери не е директно достъпна, а само през междинния слой [4, 5].

Приложение на трислойната архитектура в разпределени системи за автоматизация

В трислойната архитектура за разпределена автоматизация, както и в стандартната има различни реализации на междинния слой. Всяка от тези реализации решава различни проблеми на стандартната двуслойна архитектура. Междинният слой може да поддържа и пренарежда опашки от транзакции към слоя с данни, да синхронизира достъпа до разпределени данни, да изпълнява бизнес логиката на системата, да формализира резултата от заявките към слоя от данни, така че да е удобен за визуализация и други [7, 8].

1. *Модел със сървър за транзакции.*

В този модел клиентската програма се грижи за формиране на заявките към слоя от данни и визуализацията на получените резултати. Междинният слой поддържа опашка за входните заявки от клиентите и за резултатите, върнати от мрежата от контролери. Негова грижа е да пренасочи всяка заявка към съответния контролер от мрежата, който може да я удовлетвори. Също така той заменя еднотипни заявки от различни клиенти към един контролер с една заявка и после мултиплицира отговора и го пренасочва към съответните клиенти [1, 7, 8].

С тази функционалност на междинния слой се постига намалена натовареност на мрежата от контролери чрез групирането на заявки. Друго предимство е възможността заявките в опашката на сървъра да се обработват по приоритетна схема. По този начин се постига предвидимост на времената заявка/отговор, което е ключово в автоматизираните системи [1, 7, 8].

2. *Модел със сървър за отдалечени услуги.*

Този модел се базира на представянето на функциите на всеки възел от мрежата от контролери като услуги. Функциите на всеки контролер се привързват към софтуерни компоненти на сървъра от междинния слой. Така клиентът работи с компонентите на сървъра, а не директно с контролерите. Моделът е показан на фигура 3 [4, 5].

Фигура 3: Трислоен модел със сървър за отдалечени услуги.

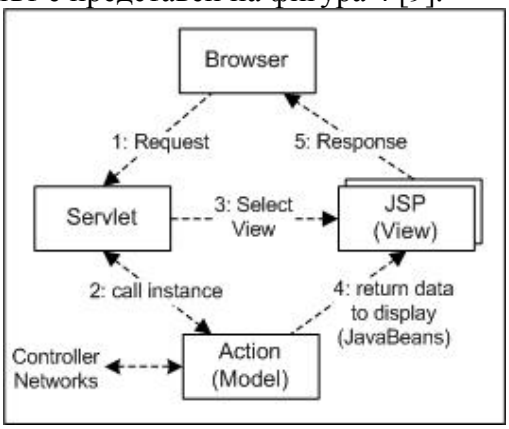
Използването на компоненти, свързващи се с контролерите скрито от потребителите и възможността да се регистрират динамично контролери в междинния слой, правят този модел добре защитен и много гъвкав. Недостатъците му са свързани с трудността на управление при голям брой контролери и липсата на стандарти за описание на услугите, представящи контролерите [4, 5].

3. *Модел с Web сървър.*

Това е една от най-разпространените реализации на трислойния модел. Понякога е наричан четирислоен, тъй като съдържа четири компоненти. На клиентския слой работи стандартен интернет браузер, който прави потребителския интерфейс стандартен. На слоя за данни имаме база данни или мрежа от контролери. На междинния слой съвместно съществуват бизнес логиката и презентационната логика. Така моделът се базира до голяма степен на динамични HTML документи и скриптове от страна на сървъра [1, 2, 4].

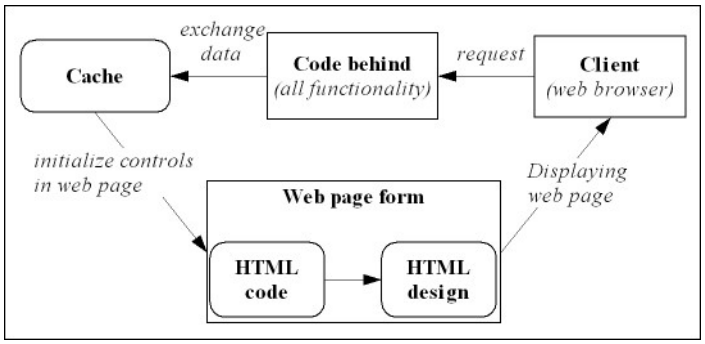
Двете най-популярни сървърни технологии за създаване на активни HTML документи са: ASP (Active Server Pages) [10] на Microsoft и JSP (Java Server Pages) [9] на Sun.

В модела на JSP се прави разделение между подаването на заявката, изпълнението на функционалността и представянето на резултата. Заявката на клиента се приема от Servlet, който извършва предварителна проверка на данните, след което ги предава към JavaBean, който извършва комуникацията с контролерите и получените резултати предава на JSP страница, която се грижи основно за визуализацията на резултата. Предимството на този модел е, че разделя презентационната от бизнес логиката, както и създаването на капсулирана функционалност (JavaBean), която може да се преизползва. Моделът е представен на фигура 4 [9].



Фигура 4: Трислоен модел с JSP.

Технологията на ASP.NET комбинира скриптове, изпълнявани при клиента (Jscript, VBScript) , HTML(XML) тагове и код (написан на всеки програмен език от .NET платформата), изпълняван на сървъра в един документ. Диаграма на ASP.NET приложение е показано на фигура 5 [10].



Фигура 5: Трислоен модел с ASP.

Заявката от клиентски браузър се обработва като се стартира функция, която зарежда страницата, но така нареченият Code Behind извършва цялата реализирана функционалност. Изтеглят се данните от контролерите, след което се прехвърлят в реализирания от .NET средата кеш. По този начин данните в кеша са достъпни от всички форми на приложението. Именно по описания начин се инициализират контролите на HTML страницата, след което тя се визуализира [10].

4. Трислоен модел с Web услуги.

Този модел залага на универсалността на описанието на данните чрез XML и универсалността на предаване им чрез HTTP. Подобно на модела, представен в точка 2, функционалността на мрежата от контролери се представя като набор от услуги, но тук описването на услугите е на станадартен език – WSDL. Тези услуги са динамично откриваеми посредством регистрирането им в директории за услуги – UDDI, ebXML [6].

Този модел се използва в комуникацията между програми през Интернет поради независимостта от мрежовата апаратура по пътя между клиента и доставчика на Web услугата [2, 6].

Тъй като контролерите в общия случай са с ограничени програмни ресурси, интегрирането им към архиректурата на Web услугите е посредством междинния слой. Той предлага услуги, отговарящи на функционалността на всички възли от мрежата от контролери [2].

Изводи и бъдещо развитие

Настоящата статия прави преглед на някои популярни технологии в разпределените бизнес системи и показва възможността за трансфера на тези технологии в разпределената автоматизация. Увеличаването на изчислителните ресурси на вградените системи и намаляването на цените им позволява все по-успешния трансфер на технологии.

Автоматизация през последните години е в процес на бурно развитие, преминавайки от специфичните индустриални протоколи към стандартните решения в Интернет. Това позволява синхронизацията на програмните системи за управление на производството с информационната система на дадено предприятие, независимо от физическото разположение.

Благодарности

Изследванията в настоящата работа са финансирани от Фонда за научни изследвания към МОН – проект “ВУ-966/2005”, “Web Services and Data Integration in Distributed Automation and Information Systems in Internet Environment”, договор “ВУ-МИ-108/2005”.

Литература

1. Spasov, G. , N. Kakanakov, “*CGI-based applications for distributed embedded systems for monitoring temperature and humidity”*, Proceedings on the International Conference – CompSysTech‘04, 17-18 June 2004, Rousse, Bulgaria, pp. I.6-1 – I.6-6.
2. Kakanakov, N., G. Spasov, “*Adaptation of Web service architecture in distributed embedded systems”*, Proceedings on the International Conference – CompSysTech‘05, 16-17 June 2005, Varna, Bulgaria, pp IIIB.10-1 – IIIB.10-6.
3. Borriello, G., R. Want, “*Embedded Computation Meets the World Wide Web*”, Communications of ACM, Vol. 43 №5, May 2000, pp. 59-66.
4. Topp, U., P. Müller, *"Web based service for embedded devices"*, 2001
5. Jazdi, N., *"Component-based and Distributed Web Application for Embedded Systems"*, International Conference on Intelligent Agents, Web Technology and Internet Commerce - 2001.
6. Kreger H., “Web Services Conceptual Architecture (WSCA 1.0)”, IBM Software Group, May 2001, www.redbooks.ibm.com.
7. Edelstein, H., *"Unraveling Client Server Architectures."* DBMS Vol 7; N 5, p. 34, 05.1994, ISSN 1041-5173
8. Depledge, N., W. Turner, A. Woog, *"An open, distributable, three-tier client-server architecture with transaction semantics"*, Digital Technical Journal, v.7 n.1, p.34-42, Jan. 1995.
9. http://java.sun.com/products/jsp/ - JSP homepage.
10. http://www.asp.net/ - ASP.NET.

За контакти:

Инж. докторант Николай Каканаков, тел: 032 659758, e-mail: kakanak@tu-plovdiv.bg, web: http://net-lab.tu-plovdiv.bg/, Технически Университет – София, филиал Пловдив.