НАУКОВО-ТЕХНІЧНИЙ ЖУРНАЛ *ISSN 1681-4886*

4 (95)' 2012

Виходить 6 разів на рік

Видається з 23 квітня 1996 р.

ІНФОРМАЦІЙНО -КЕРУЮЧІ СИСТЕМИ НА ЗАЛІЗНИЧНОМУ ТРАНСПОРТІ

INFORMACIJNO-KERUÛCI SISTEMI NA ZALIZNICNOMU TRANSPORTI

Видання

Державної адміністрації залізниць України

Української державної академії залізничного транспорту

Зміст – Содержание – Contents

В номере опубликованы материалы выступлений участников 25-й Международной конференции "Перспективные компьютерные, управляющие и телекоммуникационные системы для железных дорог Украины"
(г. Алушта, Крым, Украина, 24-29 сентября 2012 г.)

Кривуля Г. Ф., Павлов О. А., Власов И. В.
Применение мемристорной логики в компьютерной инженерии 3

Смирнов А. А., Даниленко Д. А., Мелешко Е. В.
Метод обнаружения вредоносного программного обеспечения. Часть 1. Корреляционный анализ сетевого трафика...8

Гончарова Л. Л.
Організація комп’ютерних засобів управління технологічними процесами електропостачання 15

Жученко О. С., Суєта О. В., Соловйов А. О.
Оцінка кількості потоків пакетів у кільцевій мережі на основі обладнання третього рівня...............................20

Бочаров А. П., Карбивский Ф. А., Пасечник В. И.
Корпоративное управление и трансформация бизнес-модели при реформировании железнодорожного транспорта Украины: проблемы и пути решения.......24

Соловьев В. М., Сперанский Д. В., Жничков Р. Ю.
Платформа управления виртуальной вычислительной средой..36

Сытник В. Б.
Адаптивная фильтрация и дифференцирование сигналов в системах автоматического управления 42

Нічога В. О., Сторож І. В., Ващишин Л. В.
Аналіз сигналів деяких небезпечних дефектів при магнітній дефектоскопії залізничних рейок за допомогою віконного Фур'є та вейвлет – перетворень 46

Пшинько А. Н., Скалозуб В. В., Жуковицкий И. В., Распопов А. С.
Железнодорожные интеллектуальные транспортные системы и концепция международной программы подготовки магистров в области ИТС CITISET........ 52

Замула А. А., Землянко Ю. В.
Оценивание временной задержки сигнала с использованием технологии распределенного спектра 58

Кривуля Г. Ф., Сыревич Е. Е., Карасев А. Л.
Diagnostic data storage and processing usig axeda platform 63

Кривуля Г. Ф., Бабич А. В., Мова А. Ю.
Моделирование RTCP протокола с использованием диагностического узла в видеоконференцсвязи... 67

Мироновский Л. А.
Симметричные реализации динамических систем... 71

Пахомова В. М., Федоренко Ю. О.
Рішення задачі маршрутизації в комп'ютерній мережі Придніпровської залізниці на основі нейронної моделі Хопфілда... 76

Рязанцев А. И., Скарга-Бандурова И. С., Коваленко Я. П.
Использование методов автоматного проектирования при разработке программного обеспечения систем железнодорожной автоматизации 85

УДК 32.973.202

КРИВУЛЯ Г. Ф. (Krivulya G.F.), д.т.н., профессор,
СЫРЕВИЧ Е. Е. (Syrevitch Yev.Yef), к.т.н., доцент,
КАРАСЕВ А. Л. (Karasyov A.L.), ведущий разработчик программного обеспечения (Компании Global Logic, г. Харьков)

Diagnostic data storage and processing usig axeda platform

Introduction

Diagnosis of digital devices requires storing of tests, etalon responses, fault lists etc. This data is called diagnostic information (DI). The amount of DI is very huge for modern digital devices. That's why complex solution for storing and reduction of DI is urgent nowadays. There are few alternatives of how to store DI.

The first one is to store DI on a local machine (PC) in a database or spreadsheets. It takes a lot of space for data storage. But there are a set of approaches for DI reduction [1–5]. Mainly they are based on a *fault dictionary* (fault dictionary is a database of simulated test responses for all modeled faults) used by some diagnosis algorithms. To reduce the amount of data used for fault location, a fault dictionary does not store the entire response R_f caused by the fault *f*, but only a "signature" usually consisting of the list of errors contained in R_f. However a fault dictionary still is large and requires compressing. Compression can be applied for etalons in some cases as well. Some approaches are based on usage of DI masks. All such reductions require more sophisticated process of storing DI.

The second alternative is to store DI (in a shape of a fault dictionaries or fault tables as well) on a local server. It allows to collect information of different testing threads in parallel. In this case there must be special requirements for server performance and network topology which make DI processing and storage more expensive.

The third alternative is to use a "*cloud*" platform. It includes all the advantages of a local server and even more such as remote access to the storage, simple processing mechanisms and platform management, easiness in increasing the performance on demand, less cost and so on. From the other hand this alternative requires equipment to be connected to a network, able to send/receive information and perform commands.

This article is the first one in a series and it describes *Axeda cloud platform* to be used for digital device testing and diagnosing, its features and capabilities as well as its internal data model. Moreover it shows how to use Axeda data model to take more advantage of it

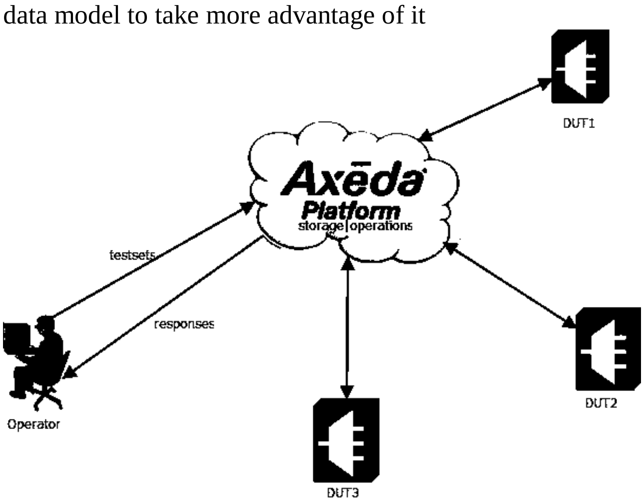

Figure 1 – Diagnosing with *Axeda cloud*

Axeda platform overview

The Axeda Platform is a secure and scalable foundation on which to build and deploy machine to machine (M2M) applications for connected products. The Axeda Platform manages the communications between remote devices, systems, and businesses, as well as live and historic data and alarm information for devices. The Platform operates as a central repository for Axeda "objects" and their relationships, current state, assignments, and functionality.

The Axeda SDK provides an application programming interface (API) to the Platform operations and objects. Using the Java-based API, you can perform operations against the Platform. Your custom code can manage, locate, and operate on Platform objects.

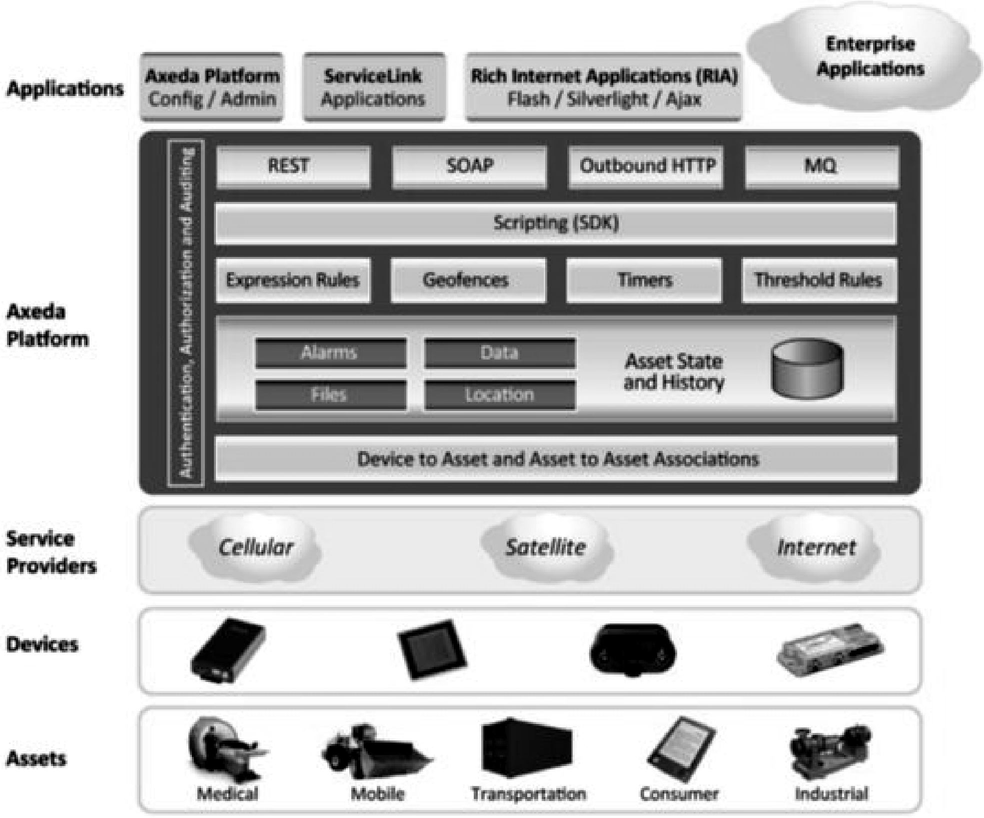

Figure 2 – Axeda Platform

The Axeda Platform includes built-in Web Services and other technologies for application development, core infrastructure capabilities, and support for leading wireless communications, enterprise applications, and edge devices.

It's not always possible to rely on being able to connect to a device when you need data, so the platform stores all the data required for each device, and the application can rely on a connection to the platform.

Connected products likely come in a wide variety of models and configurations, so Axeda provides a simple way for you to define a data model for the application. The model can include the different types of data that a connected product generates, including location data and alarms. The platform combines built-in functions and support for custom application logic.

The platform includes a Groovy Java development environment and runtime for when it's needed to extend its built-in functions. It's also possible to use Groovy to define a Web Service interface for client applications. Another advantage of the Axeda cloud platform is that it allows to configure and administer it using a simple browser-based interface or through Web Services.

Axeda Data Model

The Axeda Platform has a mature data model that's important to understand when planning to build applications. First, this will introduce the existing objects and how they relate to each other.

- *Model* – the definition of a type of asset. The model consists of a set of dataitems (its inputs and outputs) and alarms. The platform applies logic to a model, so as assets grow, the system is scalable in terms of management.
- *Asset* – or sometimes called Device. An asset has an identifier called a Serial Number which must be unique within its model. Agents report information in terms of the asset. Logic is applied to data and events about that asset.
- *Dataitem* – a named reading, such as a sensor or computer value. Dataitems are timestamped values in a sequence. For example, hourly temperatures, or odometer readings, or daily usage statistics. The number of named dataitems is unlimited. Dataitems can be written as well as read, so a value can be sent to an "output". A dataitem can be a Digital (Boolean), Analog (real value) or a String.
- *Mobile Location* - a lat/long pair typically read from GPS. This is used to map assets as they move.
- *Alarms* – have a name, severity, description, active flag, timestamp, and optional embedded dataitem and value. Alarms sent from an agent may result from logic that detects a condition. An alarm indicates something that's wrong.
- *Files* – arbitrary files can be uploaded from an agent.

Files are often uploaded when an alarm has been raised, or on demand from a user or rule.

Axeda Platform is asset-centric. An asset is an instance of a Model. Each asset is identified by its Model and Serial Number pair. Associated with an asset are: organization, location, contacts, asset groups, properties, condition, dataitems, alarms and files.

Information is processed and organized in the context of an asset, but the processing is managed for models. The only scalable way to manage a lot of assets is to apply rules by kind of asset, not individual assets.

Rules apply logic to data as it happens. When a new dataitem is reported, a rule may check against its threshold. When an alarm is created, a rule may create a trouble ticket, or notify the user. All types of rules in the platform – Expression Rules, State Machines, and Threshold Rules – are event based.

Software packages are another entity in the Platform. Packages are used to distribute files, software, patches, etc. and to script some commands around their delivery. The deployment may be automatic or manual, to one or many.

User logins are members of user groups. *User groups* have both privileges (what they can do) and visibility (which assets they can see). User group visibility allows the group to access assets in an Asset Group, or a Model or Region.

Utilize Data Model

Dataitems can be configured to store no data, current value, or history. History is needed if you want to see the temperature plot over the last day. Many times, current value is all that's needed to process rules and see the state of an asset. The option not to store a dataitem makes sense if the dataitem is only used to run a rule, or if it will just be sent to another application. In a pass-through mode, the dataitem doesn't need to be stored at all. A similar situation is if a string dataitem is parsed by the rule calling a Groovy script.

Alarms are almost always used to notify people that they should do something. Alarms in Axeda have a lifecycle that corresponds to how people interact with them. An alarm begins its life when it's created. From that point, the alarm can be acknowledged, escalated, closed, suppressed and disabled.

Files are uploaded for a few reasons. Log files are typically uploaded so a service tech can diagnose a problem. Data files can be uploaded so a script or external system can process the file and take appropriate action. This can be another way of sending information that doesn't fit in a dataitem.

Extended Objects are attributes that can be added to the objects described here, or can be complete objects that live on their own. The application can read and write these objects or attributes, and query them.

Conclusion

The introduced platform will support different solution architectures effectively, including Rich Internet Application (RIA) clients and integration with enterprise applications. The platform also supports Axeda Service-Link remote service applications. All above can dramatically increase diagnostics speed by refusing from slow and expensive local PCs and servers.

The article propose to use the Axeda “cloud” platform for storage and processing of diagnostic information during testing and diagnosing remote digital devices of different type. An operator, dealing only with the platform, has a possibility to form tests and analyze responses without direct contact with a DUT. Localizing the information the “cloud” allows to speed-up it’s processing and release the load on local stations and servers

Literature

1. *Pomeranz I., Reddy S.M.* On the generation of small dictionaries for fault location // Proc. of the 1992 IEEE/ACM Intern. Conf. on Computer-Aided design (ICCAD ’92). Los Alamitos, CA, USA, 1992. – Pp. 272–279.
2. *Arslan B., Orailoglu A.* Fault dictionary size reduction through test response superposition // Proc. of the 2002 IEEE Intern. Conf. on Computer Design: VLSI in Computers (ICCD’02). Washington, DC, USA, 2002. – Pp. 480–485.
3. *Boppana V., Hartanto I., Fuchs W.K.* Full fault dictionary storage based on labeled tree encoding // Proc. of 14th VLSI Test Symposium. Washington, DC, USA, 1996. – Pp. 174–179.
4. *Abramovici M., Breuer M.A., Friedman A.D.* Digital Systems Testing and Testable Design. N.Y.: Computer Science Press Inc., 1996.
5. *Zhang Y., Agrawal V.D.* A Diagnostic. Test Generation System // Proc. International Test conf., 2010.

Резюме

Статья описывает возможности «облачной» платформы Axeda для хранения и обработки диагностической информации во время тестирования и диагностирования удаленных цифровых устройств смешанного типа. Оператор, взаимодействуя только с платформой, имеет возможность формировать тесты и анализировать ответные реакции без непосредственного контакта с DUT. Размещение диагностической информации в «облаке» позволяет ускорить процесс ее обработки и снимает нагрузку с локальных станций и серверов

Стаття надає інформацію щодо можливостей «хмарної» платформи Axeda для збереження та обробки діагностичної інформації під час тестування та діагностування віддалених цифрових пристроїв різних типів. Оператор,який взаємодіє тільки з платформою, має можливість формувати тести та аналізувати відповідні реакцій без безпосереднього контакту з DUT. Розміщення діагностичної інформації в «хмарі» дозволяє прискорити процес її обробки и знімає навантаження з локальних станцій та серверів

The article gives the information about possibilities of the Axeda “cloud” platform for storage and processing of diagnostic information during testing and diagnosing remote digital devices of different type. An operator, dealing only with the platform, has a possibility to form tests and analyze responses without direct contact with a DUT. Localizing the information the “cloud” allows to speed-up it’s processing and release the load on local stations and servers

Ключові слова: "облачная" платформа, диагностическая информация в облаке, "cloud" platform, diagnostic information in the "cloud"

Рецензент д.т.н., профессор Дербунович Л. В. (НТУ "ХПИ")

Поступила 12.07.2012 г.