Relevant Aspects in Model-Driven Development for Mobile Applications

Estevan R Gómez T.[1]
[1] Universidad de las Fuerzas Armadas "ESPE", Ecuador, ergomez@espe.edu.ec

Abstract– When we build software must consider that this is not an easy task, especially if its complexity is high. To succeed in developing software should have a high commitment of the development team, expensive resources, highly trained specialists and processes and increasingly formal methods. The models at different levels of abstraction arise in order to expedite the process of building software, which is focused movement on the use of models at different levels of abstraction.

There are some important proposals in this regard those as Model Driven Architecture and Model Driven Software Development .This paper reviews the literature about these two proposals. Describe the fundamental principles aimed at shaping the theoretical basis of the two proposals and the main tools that implement model-driven development for mobile application development work. Describe the fundamental principles aimed at shaping the theoretical basis of the two proposals and the main tools that implement model-driven development for mobile application development work. Also analyzed the importance of Metamodels and possible transformations to generate auto code.

Keywords— Model Driven Development, Model Driven Architecture, Mobile Applications, Context aware.

Digital Object Identifier (DOI): http://dx.doi.org/10.18687/LACCEI2015.1.1.264
ISBN: 13 978-0-9822896-8-6
ISSN: 2414-6668

Relevant Aspects in model-driven development for mobile applications

Estevan R Gómez T.
Universidad de las Fuerzas Armadas "ESPE", Ecuador
E-mail: ergomez@espe.edu.ec

Abstract–

When we build software must consider that this is not an easy task, especially if its complexity is high. To succeed in developing software should have a high commitment of the development team, expensive resources, highly trained specialists and processes and increasingly formal methods. The models at different levels of abstraction arise in order to expedite the process of building software, which is focused movement on the use of models at different levels of abstraction.
There are some important proposals in this regard those as Model Driven Architecture and Model Driven Software Development .This paper reviews the literature about these two proposals. Describe the fundamental principles aimed at shaping the theoretical basis of the two proposals and the main tools that implement model-driven development for mobile application development work. Describe the fundamental principles aimed at shaping the theoretical basis of the two proposals and the main tools that implement model-driven development for mobile application development work. Also analyzed the importance of Metamodels and possible transformations to generate auto code.

.

Keywords-- Model Driven Development, Model Driven Architecture, Mobile Applications, Context aware.

I. INTRODUCTION

It is critical to consider how the constant progress of technological developments have changed the traditional way of performing certain daily habits. One example is the use of the phone; far has left the notion of simply use your phone to make calls and today has become a device with computing power, considered an indispensable means of contact, information and entertainment

There is great (ever growing) number of applications for mobile devices that give them endless useful and convenient services to users, allowing them access to any content anywhere and maintaining communication with others in the most varied forms, breaking geographical barriers and time. Following this new concept in mobile services, arises the well known mobile computing, ubiquitous features -always is available at any moment, custom-each user consumes and suitable information according to their interests, geolocation -capable of the location of users to provide products and services acording-, context sensitive, among others. In this sense, any information that characterizes the situation in which the user interacts with an application, it is called context. When we refer to context-sensitive mobile computing refers to applications that consider the contextual information from the environment where the user is located and act accordingly, providing relevant and adequate information to the context where the user is immersed done. The geographical position is a context information, perhaps the most important in this type of applications. This last feature is in full expansion giving rise to positioning based services (location-based services, LBS). Positioning based services refer to applications using the geographical position of a mobile device to provide services based on such information. Such services include assistance in navigation, identification in case of emergencies or disasters, social networks to find friends, location of points of interest in maps and more.

To successfully use the LBS services requires proper positioning technology, geographic information system to identify areas and fundamentally interoperable mobile devices and adequate network infrastructure that enables connection wherever the user is.

Driven development model aims to free the engineer of the technological aspects (programming) to center in the engineering (architecture, modeling and design). In a similar manner as COBOL freed to programmers of the minutiae of Hardware [1]

The current trend is that systems are becoming more complex and heterogeneous, further each day the number of users of mobile devices increases, which demand high efficiency in response times, and applications that enable them to carry out their activities through the re-use of resources, robust and adaptive systems.

Portable devices increasingly used in a wide range of mobile applications orientation. Typical examples are the guides of urban areas, museum guides and aids exhibitions. The application refers to the provision of specific services of the context in which the context is typically identified by combining data on the location, time, user profile, the device profile, network conditions and usage scenario [74]

Reference [74], defined context as any information characterizing a situation concerning the interaction between users, applications and the surrounding environment.

Nowadays, the advance of communications networks globally contribute to the way people communicate, do business and innovation of learning methods, thus it is transforming the world and getting closer to the people, through the innovation in global communications; this makes changes occur in all areas of human activity, eg competitiveness, product exchange, trade agreements between countries, employment and quality of life.

13th LACCEI Annual International Conference: "Engineering Education Facing the Grand Challenges, What Are We Doing?"
July 29-31, 2015, Santo Domingo, Dominican Republic **ISBN:** 13 978-0-9822896-8-6 **ISSN:** 2414-6668
DOI: http://dx.doi.org/10.18687/LACCEI2015.1.1.264

The use of mobile applications contribute to removing barriers of approaching knowledge and technological advances make an easy way to reach a wider audience; For this reason, the Internet has penetrated all spheres of society, revolutionizing the way the information is received and involving more people in the use of them; as a result there is more business opportunity.

The desire to have access to information from anywhere, anytime, and through a mobile device, it is a reality. The global Internet economy, built around a desktop, now expands to a galaxy of smaller, simpler devices: cell phones, smartphones, PDAs, handheld devices, information systems and other devices in vehicles access at home.

These factors make it imperative that a development model for mobile applications is coherent, consistent, and versatile.

The first attempts to use the models in the development process began with diagrams proposed in structured analysis and design [2] This practice evolved with the emergence of Object-oriented analysis and design and the subsequent introduction of the Unified Modeling Language (UML) [19], [20]. Though, the first practices modeling were guided primarily to document the system itself. The notion of software development driven models, is a true milestone in the history of software development, as it aims to incorporate the models as an essential part of the entire development process.

Leading promoters of this idea are the Object Management Group (OMG) [8] and a global movement of theoretical and practical researchers grouped into what is called "Software Development Guided by Models" MDSD also known as "Model-Driven Development" MDD [8], [9].The fundamental principle of these two line of thought is to use models to develop software, and its main purpose is to build practical tools and concepts leading to the automation of software development.

For implementing the principle of abstraction levels, MDA has identified three models to describe the system [15]: Independent Computer Model CIM, Platform Independent Model PIM and Platform Specific Model PSM. Level of abstraction is a CIM and it represents the business model from the point of view of the user. PIM is a conceptual model that represents the point of view of the user interpreted by the software engineer. PSM is a model of the writing system in order to move to a specific technology platform. By their hand OMG , has developed a set of standards, evolving, called MDA [21] " Model Driven Architecture" The fundamental principles of MDA for building software are: use of models throughout the development process definition a model for each level of abstraction and model transformations. These principles help automate the software development process holistically, from its initial stages to obtain executable code.

Often traditional computing applications are static and inflexible.

They are designed to run on a specific device, providing a set of predetermined user and have integrated functions contextual dependencies on them. Such application models are not adequate to operate in a ubiquitous computing environment, which is characterized by its rich context, mobility of users and devices (PDAs, smartphones, ...) and the appearance and disappearance of resources over time

Today, where ubiquitous (pervasive) computing, based on sensitivity to context takes an important place in everyday life, the development of context-aware applications that provide appropriate services for users is necessary, considering its multiple contexts

II. DIFFERENCE BETWEEN MDA AND MDSD

A. General characteristics

Although MDA and MDSD are different communities share the same purpose: the use of models for developing software. While MSDS deals with models using software development, MDA focuses on defining standards for software development models intensive use. You could say that MDA is the standardization of MDSD

The difference between MDA and MDSD has been shown by several authors and researchers. According to [14] suggests that MDA can be seen as a set of standards whose practical application is the development of model-driven software, and MDSD main activity is the construction of tools that favor the development by using models. According to [64] states that methods and tools that support both MDA as MDSD should specify automatic transformations between different models, differentiating the two terms

Referring to [27], suggests that model driven development of software, has to do with the models used by architects and programmers to feed code generators are techniques that are also under the framework of MDSD, but can´t be properly called MDA.

In conclusion, although MDA and MDSD are different approaches, both privilege the use of the model as development language'. MDSD has existed much earlier than MDA and development practice as research to generate methods, techniques, tools and theoretical constructs based on models. MDA for its part has made a great effort to standardize the wide but patchy activity driven software development models MDSD

B. GENERAL THEORETICAL FOUNDATIONS

MDA and MDSD are complementary approaches using the model as a guideline for development. MDA to be a set of standards promoted by OMG, is characterized by strong theoretical bases, while MDSD for being an initiative in the hands of a large research community, focuses primarily on the development of tools that facilitate the automation of development. At the same time, reducing development budgets and a difficult economic environment necessary to make highly efficient use of resources for development. With designs increasingly complex integrated products and reduced

product life cycles, efficient development has become essential. The emergence of model-driven development (MDD) has made it possible to accelerate the development process. With the model-driven development (MDD) software engineers can understand and analyze more clearly the needs, define design specifications, test case generation system and automatic code generation

C. MDA IN THE CONTEXT OF STANDARDS OMG

In this environment, platform independent models that include OMG modeling standards can be developed using open or proprietary platforms such as CORBA, Java, .NET, XMI / XML and web-based platforms (see Fig 1).

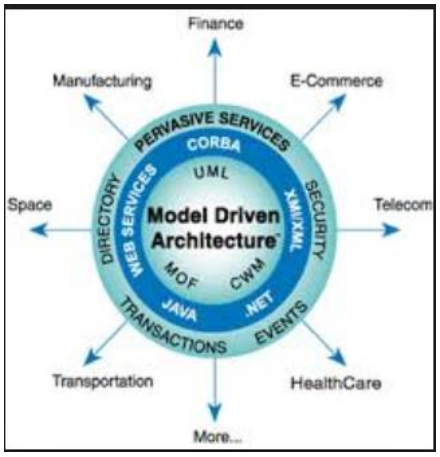

Fig. 1 Model Driven Architecture(OMG).

However, the dominant trend for dependent platform model is the use of java, because the resources and facilities of this platform. Additionally, as new technologies emerge, MDA quickly implements the relevant specifications, so as to facilitate the integration process. It is concluded that MDA is more than a middleware provides a structured and comprehensive solution for interoperability and portability of applications in the future [29], [30].

Standards that form the core of MDA, the Common Warehouse Model CWM requires special attention because it provides guidelines for data management and integration, as shown in the next section.

III. COMMON WAREHOUSING MODEL CWM

It is an OMG standard for data warehousing, defining the specification for metadata modeling, non-relational, multidimensional, relational and other objects of the data warehouse environment. It covers the entire lifecycle of design, construction and management of applications and management support life cycle [13].

CWM specifies the interfaces that enable metadata exchange of data warehousing and business intelligence from data warehousing appliances, in heterogeneous distributed environments, such as tools, platforms and repositories [11], [12], [13].

Fig. 2 Model Driven Architecture (OMG).

CWM is based on three standards that forms the core of the architecture of the OMG metadata repository [14].

Unified Modeling Language (UML). Primarily graphical language for defining and representing models and meta-models of the system design models and information models.

• Meta-Object Facility (MOF). It is a standard that defines metadata models and models of models. Provides tools with programmable interfaces for storing and accessing metadata in a repository.

• XML Metadata Interchange (XMI). Standard for exchanging metadata submitted as files in XML format.

A CWM specification includes basic components and a set of metamodels. The basic components of CWM are shown in Figure 2 and are as follows: CWM metamodel, metadata interchange format for shared warehousing (CWM DTD) interchange format for the metamodel (CWM XML) and API access to metadata shared warehousing (CWM IDL) [13]

IV.META MODELS

The metamodel is the architecture of the conceptual information that classifies information we can use to build solutions, understanding problem domains, and create practices that ensure we build the system we build.

The definition of a "model solution" is actually a symptom of why modeling approaches have not reached the level of care they deserve.

The way to model the problem may be related to how the model of the functional specification and yet not related to how to build the solution. The modeling is good, but too much of it can be difficult to get everyone involved.

That's why "focus on the metamodel with which it will build the solution." This is the greatest ROI. You can use it to create a model solution, but it may be too much work. What is

it trying to carve out a pragmatic path to successful model driven engineering, without eliminating the modeling.[1]

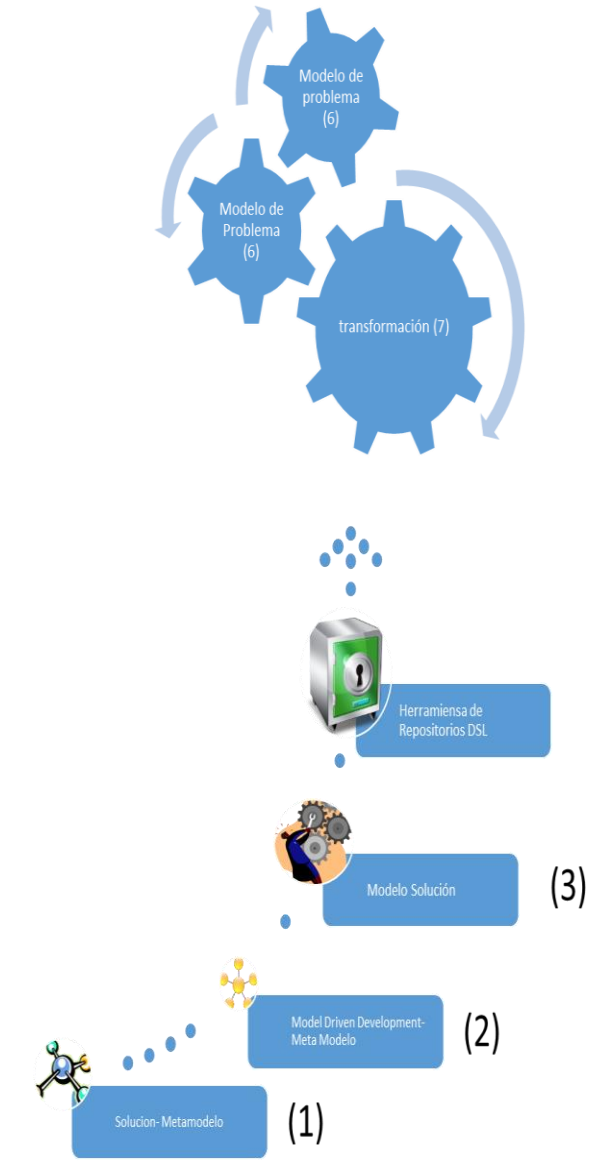

Fig. 3 Basic Steps

Basically, what is proposed is that if a solution metamodel that includes elements of "application" (containing the solution behavior in relation to the metamodel) essentially has "scrambled" created the model solution (for free). The application element must not contain any invocation or platform specific library. This is the key point you have to understand, if you create a metamodel without implementation elements and try to save the metamodel own behavior is most likely not going to work well.

Then and only then, you should begin to understand how (and) can model the problem domain and possibly tying them with the solution model. Successful model-driven engineering (steps 1 to 7) approach is then proposed:

A. ISSUES IN THE TRANSFORMATION IN THE APPROACHES FOCUSED ON MODELS

The transformation of models allows developers to build models with the problem representation and transform on up to a computer system that works. This promise made in the most recent methodological approaches greatly enhances the possibilities of reusing elements of software development, such as objects, components, patterns and frameworks [44] for almost automatic generation of software applications. A typical transformation strategies is based on building a system based on a metamodel of origin that apply transformation rules defined model, and transformed into a model that is based on a target metamodel [18]

V. MODEL DRIVEN DEVELOPMENT AND ITS APPLICATION TO MOBILE APPLICATION DEVELOPMENT

The software for Android is rapidly gaining market share with their applications for a variety of devices such as smarthphones, tablets, televisions and entertainment systems. [1] The variety of market segments, an increase in the number of new devices and demand for different users are forcing device manufacturers and application providers to introduce innovative high quality products in shorter timeframes.

PC WORLD says "Starting in May, a new chip install this platform will allow more types of personal devices[2]

Competition from Android continues to expand. Far from settling for the market of smartphones, starting next May, an embedded within other personal devices, chip enable install the operating system Google. These devices find from media players to MID (Mobile Internet Devices or internet access devices) through drivers for leisure or medical monitoring tools.

This is a chip developed by the company that makes installing Embedded Alley Android based Au1250 chip, developed by RMI systems. Specifically managed to carry the current mode of implementation of Java engine that incorporates Android (instead of having the JME standard has its own virtual machine, called Dalvik) to MIPS instructions, so that they are compatible with the drivers, for example, for Nintendo Wii, etc."

[1] http://www.ebpml.org/blog/130.htm

2 http://www.pcworld.com.mx/Articulos/3984.htm

VI PROCESS FOR GENERATING APPLICATIONS ANDROID

The process should allow Android applications generate from NDT methodology. NDT define transformations that tell you how to get every model from requirements definition, ie in the first place the CIM model specification defines the requirements, then the Platform Independent Model (PIM) is designed. NDT models are defined with UML Profiles in EA modeling tool allowing the development stages including the requirements model, content model, navigation model and process model.

Once defined the PIM model transformation will take place towards the PSM model, and then a model transformation is performed text.

De facto standard language model transformations text for design is called MOF Model to Text Transformation Language (mof2text). Some advantages of mof2text language is that not only enables the generation of code for many different platforms based on the same model, but also allows the automatic generation of any textual representation of the model.

MD² = Model-driven Mobile Development

The development of the framework MD² consisted basically three steps. First overall architecture has to be specified.

Based on the architecture had to be written language. The language offers, together with a grammar that modelers must meet the metamodel application models that can be modeled with MD². This metamodel is required for the third step, which is the development of generators.

To develop generators, was chosen based on the prototype approach. This means that we first develop prototype applications directly to Android and iOS. These prototypes must be representative and comply with the overall architecture. Later these prototypes will be used to develop generators. Therefore, we first analyzed the prototypes of interdependent parts of a given model and will always be the same. We move these parts to libraries. The other parts build the basis for the development of the generators. We take part after another and transform them into buildable code by identifying parts depending on the model and allow them SHALL be generated based on the input model.[3]

VII DEVELOPMENT TOOLS

Tools that can be used are Eclipse to develop Android applications, and Xcode [4] for iOS Apps, you can also use the Framework of NDT

The advantages of Xtend [5] are better integration with Xtext [6] framework that is more flexible because the templates and behavior are handled in the methods and Xtend uses a Java like language that offers many features to facilitate construction of the generator. Disadvantages are no support for handling generated code using a general purpose language that leads to a frame not standardized.

The decision to be taken is the tool to develop generators. They have two options. Conveniently we could use Xtend or Acceleo[7].

The benefits of Acceleo on the other hand are a good integration into Eclipse and support development functionality such as debugging, tracking and tracing, support for protected areas that allow easy manipulation of the generated code, construction of the native generator and MOF model applies to standard text transformation as specified by the OMG. Acceleo disadvantages are that it is static and which is based on a template and is difficult to integrate into a multi-environment model.

VIII A METHODOLOGICAL PROPOSAL NDT

It is based on a set of metamodels, which are transparent to the development group, which supports the development process.

Take care of requirements traceability from capture to analysis, providing a systematic development process based on formal transformations described with QVT leading to implementation.

Working with UML, is a proposal that can be easily incorporated into other methodological environments such as Metric.

NDT has had and is having a great practical applicability in companies and real projects.

In recent years, the NDT methodology has evolved to permit their use in practical environments, being now one of the best methodology to address the development of any software project proposals, particularly software projects oriented to WEB.[8]

A. LIFE CYCLE OF NDT

Android Generate code to benefit from the proposed methodology NDT, with NDT-Tool Suite [9] (Currently Enterprise Architect) help improve productivity and time to market and reducing development costs. NDT provide a very comfortable environment, thereby obtaining a unique relationship model-code NDT-Android. Using the methodology connect the requirements, design and code also

[3] http://wwu-pi.github.io/md2-web/res/MD2-Documentation.pdf

[4] https://developer.apple.com/xcode/

[5] http://eclipse.org/xtend/

[6] https://eclipse.org/Xtext/

[7] https://eclipse.org/acceleo/

[8] http://www.iwt2.org/web/opencms/IWT2/ndt/?locale=es

[9] http://www.sparxsystems.com/products/ea/12/index.html

thanks NDT-Suite also can automatically generate documentation and testing.

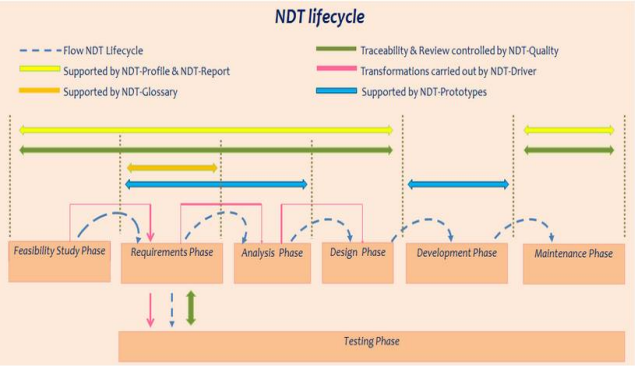

Fig. 4 NDT LIFECYCLE [10]

This initiative opens a new line of research in the IWT2 group (Engineering and Testing Web and Early) which is aligned with current trends in international research framework. NDT is today a development methodology internationally recognized and also very applied in the business world, as has been assumed by different companies.

B. Details of NDT and Android.

NDT is a proposal supported on the paradigm of model-driven engineering[11] (MDE) and provides a useful and simple environment that is based on a set of metamodels, which are transparent to the development group, which supports the process

C. Methodology NDT

The NDT process focuses on a detailed engineering phase requirements guided by objectives, which includes both capture, as the definition and verification requirements [8]. The process starts by defining the objectives and based on these, a process that can capture and define the different system requirements described.

These are classified and treated depending on the type to which they belong.

D. Requirements Division with NDT.

Storage requirements, functional actors, and non-functional interaction. Once validated these requirements, the NDT process aims to generate three models: the conceptual model, represented by a class diagram the static structure of the system; navigation model that represented by a set of class diagrams how you can navigate the system; and abstract interface model, which through a series of assessable prototypes allow to show how it will interact with the system.

The highlight of the process proposed by NDT feature is that the transition from requirements specification to these models is done systematically and independently.

It is a systematic way because NDT defined algorithms that tell how to get each model from requirements definition. And it is independent because, although there are relationships among model, a fact that is unavoidable since all refer to the same system, it is not necessary to achieve the conceptual model to get the navigation model or abstract interface. Three processes from the requirements that allow achieving these three models are defined. These models are achieved in a systematic way are called basic models. The highlight of the process proposed by NDT feature is that the transition from requirements specification to these models is done systematically and independently.

It is a systematic way because NDT defined algorithms that tell how to get each model from requirements definition. And it is independent because, although there are relationships among model, a fact that is unavoidable since all refer to the same system, it is not necessary to achieve the conceptual model to get the navigation model or abstract interface. Three processes from the requirements that allow achieving these three models are defined. These models are achieved in a systematic way are called basic models. Thus the conceptual basic model, basic navigation model and the basic model will abstract interface. These basic models should be studied by the group of analysts and may be changed if deemed appropriate.

However, a change in any of these models can be the source of an error or inconsistency committed during the requirements engineering can generate changes in other models. Therefore, after the generation of the basic models, NDT offers a guide with all the changes that can be made and to what extent they affect other models of the system or the definition of requirements.

E. Influence of Android

The power of the Android application framework that lies in the way of translating the Web on mobile applications [2]. This does not mean that the platform provided a powerful browser, which is limited to JavaScript and server-side resources, but is the basis of their performance and user interaction with the mobile device are focused on navigation models. How to navigate a mobile user in the platform is critical to your business success. The platforms that reproduce the desktop experience on a mobile device are accepted by the users. To facilitate code reuse and streamline the development process, Android applications are based on components.

The components can be of 4 types:

Activities. They are visual interfaces expecting some user action. The visual content of each activity is provided by a series of objects derived from the View class. Android provides many of these predesigned as buttons, switches, menus objects.

[10] http://www.iwt2.org/web/opencms/IWT2/ndt/?locale=en

[11] http://www.sparxsystems.com/bin/MDA%20Tool.pdf

Services. Services do not have GUI. An example would be playing a song. For a rendering application we could have several activities to show playlists or player with buttons, but the user will expect the song to keep still ringing when exiting the application (end activity), so that this application must control a service so the music is played.

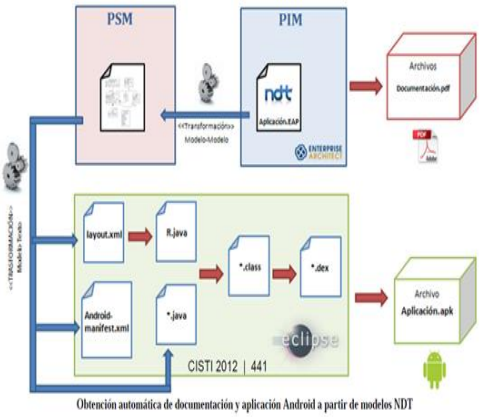

Fig. 5

Event Receivers. These components are simply listening to certain events occur (low battery, change the language of the device, downloading a new image ...)

Content providers. Allows an application to put certain data available to other applications.

In addition, all Android applications must have a AndroidManifest.xmldonde file all application components as well as the permissions required, or librearías resources and uses are defined.

NDT define transformations on how to:

Get every model from requirements definition, ie in the first place the CIM [12]model specification requirements, then the Platform Independent Model (PIM) is designed.

NDT models are defined with UML Profiles in EA modeling tool allowing the development stages including the requirements model, content model, navigation model and model and processes. Once defined the PIM model transformation will take place towards the PSM model, and then a model transformation is performed text.

E. Conclusions

1) It will be necessary to choose the tool to achieve efficient development and allows the use of Paradigm Model Driven Development.

2) To create Android applications will be essential to work with the definition of: Activities, Services and Event Receivers

3) To develop a new project is necessary to define a team which will work in parallel on Android Apps, like iOs.

4) Although the models centered approaches are still maturing, organizations are beginning to reap real benefits and manufacturers are showing their interest in this area, so that the tools are getting better. This situation has led to models centered approaches are being used on several fronts

5) NDT emerges as valid to consider for automating transformations alternative because it has a full suite that can be used in future research

[12] http://www.sparxsystems.com/bin/MDA%20Tool.pdf

REFERENCES

[1] FRANKEL, David. Model Driven Architecture -Applying MDA to Enterprise Computing. USA:Wiley Publishing, Inc. 2003.

[2] LAUDON, Kennet and LAUDON, Jane. Sistemas de información Gerencial: Administración de la Empresa Digital. Pearson Ed. 2008.

[3] Beery D. Holstein, "Speed delivery of Android devices and applications with Model-Driven development," Senior Manager, Rhapsody Product Development, IBM, 06-Jun-2011

[4] Deepak. ALUR, John. CRUPI, and Dan. MALKS, Core J2EE Patterns: Best Practices and Design Strategies, Sun Microsystems Press, 2001.

[5] Wolfgang. EMMERICH, Mikio. AOYAMA, J. SVENTEK. The impact of research on middleware technology. ACM SIGSOFT Software Engineering Notes, Vol. 32, No. 1, 2007.

[6] ROSS, Jeanne; WEILL, Peter; ROBERTSON, David. Enterprise architecture as strategy: creating a foundation for business execution. Harvard Business Scholl Press, 2010.

[7] MENTZAS, Gregoris y FRIESEN, Andreas. Semantic Enterprise Application Integration for business processes. Service-Oriented Frameworks. USA: Business Sdence Reference IGI Global. 2010

[8] OMG Object Management Group. Model-Driven Architecture Home Page, 2011 , http://www.omg.org/mda/index.htm

[9] POOLE, John. Model-Driven Architecture: Vision, Standards and Emerging Technologies, 2001 ,http://www.cwmforum.org/Model-Driven+Architecture.pdf

[10] MILLER, Frederic; VANDOME, Agnes. Common Warehouse Metamodel CWM. VDM Verlag, Alphascript Publishing, 2010.

[11] CWMFORUM. What is the Common Warehouse Metamodel CWM, 2011, http://www. cwmforum.org/about.htm

[12] KIMBALL, Ralph y ROSS, Margy. Model-Driven Data Warehousing. New York: John Wiley & Sons Inc. 2011.

[13] BARRY, Douglas K. Common Warehouse Metamodel CWM, 2011,. http://www.service-architecture.com/web-services/articles/common warehouse meta-model cwm.html

[14] POOLE, John; CHANG, Dan; TOLBERT, Douglas y MELLOR, David. Common Warehouse Metamodel Developer's Guide. New York: John Wiley & Sons Inc. 2008.

[15] AKHTER, N. y TARIQ, N. Comparison of Model Driven Architecture (MDA) based tools. Estocolmo: Institute of Technology-Karolinska University Hospital, 2005. [Links]

[16] ANDROMDA. 10 steps to write a cartridge]. 2006 http://andromda.org/index.php?option=com_content&view=category&layout=blog&id=35&Itemid=77[consulta: 06-07-2010] [Links].

[17] BALMELLI, L. et al. Model-driven system development. IBM System Journal, 2006, vol. 45, núm. 3, pp. 569-585. [Links]

[18] BÉZIVIN, J. In search of a basic principle for model driven engineering. Upgrade. 2004, vol. 5, núm. 2, pp. 21-24. [Links]

[19] BOOCH, G.; ROMBAUGH, J. y JACOBSON, I. El proceso unificado de desarrollo de software. Madrid: Addison Wesley, 1999. [Links]

[20] BOOCH, G.; ROMBAUGH, J. y JACOBSON, I. El lenguaje unificado de modelado. Madrid: Addison Wesley, 2002. [Links]

[21] CALIC, T.; DASCALU, S. y EGBERT, D. Tools for MDA Software Development: Evaluation Criteria and Set of Desirable Features. 5° International Conference on Information Technology: New Generations, Reno, CSE Department, University of Nevada, 2008. [Links]

[22] CHITCHYAN, R. et al. Survey of analysis and design approaches . AOSD, 2005. http://www.aosd-europe.net/deliverables/d11.pdf [consulta: 26012010].
[23] CZARNECKI, K. y HELSEN, S. Feature-based survey of model transformation approaches. IBM System Journal, 2006, vol. 45, núm. 3, pp. 621-645. [Links]
[24] DEN HAAN, J. 8 Reasons Why Model-Driven Approaches (will) Fail . 2008. http://www.infoq.com/articles/8-reasons-why-MDE-fails [consulta: 19032015].
[25] DE LARA, J. AToM3 A Tool for Multi-formalism Meta-Modelling]. 2006. http://atom3.cs.mcgill.ca/ [consulta: 672010] [Links].
[26] DREY, Z. et al. Kermeta language Reference manual. s. l.: IRISA, 2010. [Links]
[27] DUMAS, M. Case study: BPMN to BPEL Model Transformation. 5th International Workshop on Graph-Based Tools - GraBaTs. Zurich, 2009. [Links]
[28] EMIG, C. ; WEISSER, J. y ABECK, S. Development of SOA-BasedSoftware Systems - an Evolutionary Programming Approach. s. l.: Universitat Karlsruhe, 2006. [Links]
[29] EMIG, C. et al. Model-Driven Development of SOA Services. s. l.: Universitat Karlsruhe, 2007. [Links]
[30] FORRESTER CONSULTING. Modernizing Software Development Through Model-Driven Development. A commissioned study conducted by Forrester Consulting on behalf of Unisys. s. l., 2008. [Links]
[31] FORWARD, A. y LETHBRIDGE, C. Problems and opportunities for model-centric versus code-centric software development: a survey of software professionals. Proceedings of the 2008 international workshop on Models in software engineering, Leipzig: Association for Computing Machinery, 2008. [Links]
[32] FRABCE, R. y RUMPE, B. Model-Driven Development of Complex Software: A Research Roadmap. En: Future of Software Engineering 2007 at ICSE. Minneapolis: IEEE, 2007, pp. 37-54. [Links]
[33] GAMMA, E. et al. Design patterns, elements of reusable object-oriented software. Boston: Addison-Wesley, 1995. [Links]
[34] GARCÍA, J. et al. Un estudio comparativo de dos herramientas MDA: OptimalJ y ArcStyler. Murcia: Departamento de Informática y Sistemas, Universidad de Murcia, 2004. [Links]
[35] GREENFIELD, J. y SHORT, K. Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools. New York: Wiley, 2004. [Links]
[36] GRÓNMO, R. y OLDEVIK, J. An empirical study of the UML model transformation tool (UMT). INTEROP-ESA. First International Conference on Interoperability of Enterprise Software and Applications, Suiza, 2005. [Links]
[37] HAYWOOD, D. MDA in a Nutshell]. En: The Server Side, 2004. http://www.theserverside.com/news/1365166/MDA-Nice-idea-shame-about-the [consulta:06-07-2010] [Links].
[38] HILL, J. et al. Magic quadrant for business process management suites [document en línea]. 2009. http://mediaproducts.gartner.com/reprints/lombardi/article2/article2.html. [Links]
[39] INFORMATION SOCIETY TECHNOLOGIES (IST). Modelplex: MODELling solution for comPLEXsoftware systems]. European Commission, 2006. http:// www.modelplex.org/ [consulta: 09-08-2010] [Links].
[40] INTERNATIONAL BUSINESS MACHINES (IBM). IBM Rational Rose XDE Modeler]. 2004. http://www.uml.org.cn/UMLTools/pdf/IB2M.pdf [consulta: 06-07-2010] [Links].
[41] JÉZÉQUEL, J. Model Transformation Techniques]. Universidad de Rennes, 2006. http://www.irisa.fr/prive/jezequel/enseignement/ModelTransfo.pdf [consulta: 06-07-2010] [Links].
[42] KING'S COLLEGE LONDON, UNIVERSITY OF YORK. An Evaluation of Compuware OptimalJ Professional Edition as an MDA tool. York, 2003. [Links]
[43] KLEPPE, A.; WARMER, J. y BAST, W MDA explained: The practice and promise of model-driven architecture. New York: Addison-Wesley, 2003. [Links]
[44] LARMAN, C. UML y patrones: introducción al análisis y diseño orientado a objetos. 4a ed. Madrid: Prentice Hall, 2005. [Links]
[45] LO GIUDICE, D. The State of Model-Driven Development], 2007. http://www.forrester.com/rb/Research/clarifying_options_for_application_development_teams/q/id/41357/t/2[Consulta: 06-07-2010] [Links].
[46] LOPEZ, H. et al. Estado del arte de lenguajes y herramientas de transformación de modelos. Montevideo: Instituto de Computación, Universidad de la República, 2009. [Links]
[47] MENS, T. y van GORP, IP A taxonomy of model transformation and its application to graph transformation. Electronic Notes in Theoretical Computer Science. 2006, núm. 152, pp. 125-142. [Links]
[48] MENS, T. et al. Applying a model transformation taxonomy to graph transformation technology. Electronic Notes in Theoretical Computer Science. 2006, núm. 152, pp. 143-159. [Links]
[49] MOHAGHEGHI, P. et al. MDE adoption in industry: challenges and success criteria. New York: Springer, 2009. [Links]
[50] MOHAGHEGHI, P. y AAGEDAL, J. Evaluating quality in model-driven engineering. Proceedings of the International Workshop on Modeling in Software Engineering 2007. International Conference on Software Engineering. IEEE Computer Society, Washington, DC, 2007. [Links]
[51] NORTHROP, L. Software product line essentials. s. l.: Software Engineering Institute, Carnegie Mellon University, 2008. [Links]
[52] OASIS. Web Services Business Process Execution Language Version 2.0]. 2007. http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf [consulta: 06-06-2010] [Links].
[53] OBJECT MANAGEMENT GROUP! Model Driven Architecture (MDA) Guide vl.0.1]. 2003a. http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf [consulta: 06-07-2010] [Links].
[54] OBJECT MANAGEMENT GROUP Common Warehouse Metamodel (CWM) Specification vl.0.1]. 2003b. http://www.omg.org/spec/CWM/L1/PDF/ [consulta: 06-97-2010] [Links].
[55] OBJECT MANAGEMENT GROUP. Meta Object Facility (MOF) Core Specification v2.0.]. 2006. http://www.omg.org/spec/MOF/2.0/PDF/ [consulta: 06-07-2010] [Links].
[56] OBJECT MANAGEMENT GROUP. MOF 2.0 / XMI Mapping, v2.1.1]. 2007a. http://www.omg.org/spec/XMI/2.1.1/PDF/index.htm [consulta: 06072010] [Links].
[57] OBJECT MANAGEMENT GROUP. Object Constraint Language v2.2]. 2007b. http://www.omg.org/spec/OCL/2.2/PDF [consulta: 06072010] [Links].
[58] OBJECT MANAGEMENT GROUP. Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification v1.1]. 2009a. http://www.omg.org/spec/QVT/1.1/Beta2/PDF [consulta: 06072010] [Links].
[59] OBJECT MANAGEMENT GROUP. Business Process Modeling Notation (BPMN) FTF Beta 1 v2.0]. 2009b. http://www.omg.org/spec/BPMN/2.0/Beta1/PDF/ [consulta: 06072010] [Links].
[60] OBJECT MANAGEMENT GROUP. OMG Unified Modeling Language TM (OMG UML), Superstructure v2.3]. 2010. http://www.omg.org/spec/UML/2.3/Superstructure/PDF/ [consulta: 06072010] [Links].
[61] QUINTERO, J. y ANAYA, R. Marco de referencia para la evaluación de herramientas basadas en MDA. Memorias del X Workshop IDEAS, 2007. p. 225-238. [Links]
[62] QUINTERO, J.; CADAVID, J. y OSPINA, C. Estudio comparativo de técnicas de modelado de negocio. En: Memorias del XI Workshop IDEAS, 2008, pp. 309-314. [Links]
[63] QUINTERO, J. y PÉREZ, J. Estrategias para la definición de una técnica de modelado para arquitecturas de referencia. Memorias del XII Workshop IDEAS, 2009. [Links]

[64] SELIC, B. The pragmatics of model-driven development. IEEE Software, vol. 5, núm. 20, pp. 10-25. [Links]
[65] SENDALL, S. y KOZACZYNSKI, W Model Transformation - the Heart and Soul of Model-Driven Software Development. Geneva: Software Modeling and Verification Lab., University of Geneva, 2003. [Links]
[66] SOMMERVILLE, I. Ingeniería de software. 7a ed. Madrid: Pearson Addison Wesley, 2005. [Links]
[67] SUN DEVELOPER NETWORK. Java Metadata Interface (JMI)]. 2002. http://java.sun.com/products/jmi/ [consulta: 06072010] [Links].
[68] THE ECLIPSE FOUNDATION. ATL User Guide]. 2006. http://wiki.eclipse.org/ATL/User_Guide [consulta: 06072010] [Links].
[69] VÖLTER, M. y STAHL, T. Model-Driven Software Development (Technology, Engineering, Management). New York: Wiley, 2006. [Links]
[70] WANG, W. Evaluation of UML Model Transformation Tools. Viena: Business Informatics Group, Vienna University of Technology, 2005. [Links]
[71] W3C. XSL Transformations (XSLT). v1.0]. 1999. http://www.w3.org/TR/xslt [consulta: 06072010] [Links].
[72] OMG. Model Driven Architecture (MDA), document number ormsc/2001-07-01, 2001.
[73] Dey, A. K. (2001). Understanding and Using Context. Recuperado el 03 de 03 de 2014, de http://download.springer.com/static/pdf/638/art%253A10.1007%252Fs007790170019.pdf?auth66=1394887856_f5f0d3c631e14d9f9dd616d5aeaa3852&ext=.pdf
[74] Lovette Tom, Eamon O'Neil. (2012). Mobile Context Awareness. New York: Springer.
[75] Samyr Vale1, 2. (2013). Model Driven Development of Context-aware Service Oriented Architecture. The 11th IEEE International Conference on Computational Science and Engineering - Workshops.
[76] OMG. Enterprise Collaboration Architecture(ECA), document 04-02-01, february, 2004.