Aleksandr NIKITIN

Saint-Petersburg State Electrotechnical University "LETI" (Saint-Petersburg, Russia)

Virtual Design-Driven Framework for Electronic Design Automation systems

Abstract. *The new architecture of Electronic Design Automation systems on the basis of network-centric computing environment is proposed. The system is based by creating in each node a middleware network module Design-Driven Framework for each design object and for the various stages of its design. The virtual clustering of these middleware DDF modules is discussed.*

Streszczenie. *Zaproponowano nową architekturę komputerowego systemu wspomagania dużych projektów "Electronic Design Automation". System basuje na tworzeniu w każdym z węzłów modułu "Design-Driven-Framework". (**Wirtualna platforma programistyczna do komputerowego wspomagania dużch projektów**)*

Keywords: Electronic Design Automation (EDA), Design-Driven Framework (DDF), network-centric architecture, virtualization.
Słowa kluczowe: wspomaganie dużych projektów, Electronic Design Automation.

doi:10.12915/pe.2014.01.20

Introduction

A developer or groups of developers have to use various tools (programming languages and programming environments, as well as debugging and testing tools) when developing a large project to implement the various faces and phases of this project.

The more this issue becomes relevant when implementing large projects of Computer-Aided Engineering (CAE), such as Electronic Design Automation (EDA) systems. Here one must use different programming paradigms and programming tools for the implementation of the various phases of the project. We offer split complex project at the following stages:

- Analysis of the problem in order to determine the type and amount of resources (both computing and telecommunications, and also creative and cognitive as well as timing schedule) required for its solution.
- Development of a model of a computing system that is able to solve the problem. In fact, it is the development of a virtual computing environment and a set of virtual instruments for solving this problem.
- Development of a method for implementing a virtual computing environment and a set of virtual instruments with existing relevant resources in the network-centric [1] computer system.
- Actualization of the virtual computing environment and a set of virtual instruments on specific sets of real data and other resources into a combined network-centric computing environment.
- Resolving the task in view using the network-centric computing environment.

Implementation of computing environment as Design-Driven Framework

We propose the implementation of the network-centric computing environments by creating in each node for each particular design object (artefact) and, moreover, for the various stages of its design, a separate middleware network module called Design-Driven Framework (DDF) [2]. The DDF must be able to perform functionality, required from EDA system, and, also, have the possibility of generation source code of DDF module and its further execution based on formal specifications of the design object (an artefact) at the appropriate level in the hierarchy. To do this there is a need to develop specialized property-oriented language of functional specifications [3], which provides a hierarchical description of the design object.

Modern realities are such that, in addition to the primary requirements for the lingware of EDA systems such as completeness of functionality, reliability, performance and user-friendly user interface, the greater prominence is given to the formerly secondary requirements, namely:

- The speed of design (engineering) and upgrades (reengineering) of software and hardware products;
- The independence of the success of the project from the level of preparation and performance of specific developers.

The property-oriented language of functional specifications has two different functions:

- Provides a transition from abstract specification to the concrete implementation of one of the possible ways of solving the problem;
- At the same time verify this transition with modeling of alleged situational development procedures.

Design Driven Framework (DDF) is the instrumental environment, which transforms and merges design framework and computing environment capabilities to the end-to-end route of computer-aided design of complex and high-tech design objects.

It provides the necessary functionality through the automated creation of design modules for each specific product and, moreover, for each phase of the design, for example:

- at the initial stage of designing forms by means of property-oriented language of functional specifications the formalized description of requirements to the design object (for example, in dialogue with the developer), then generates a new description for the next stage of design;
- on the basis of the description of the design object forms requirements for composition of services, agents, models, functions and methods that need to be involved to provide design, simulation, testing, verification functions of the specific design phase;
- based on the claims received and the availability of libraries (including those on a network) that implement various models, features and design methods, produces the required (derived) source code generation project program modules, for example, by using technologies of functional programming;
- on the basis of the description of the design object shapes the computing requirements and terms, searches (in the network, if necessary) and allocates the necessary computing resources for the generated software module;
- generates a derived design software module, than forwards it (if necessary) to a dedicated network resource with a copy of DDF installed and calls it for execution;
- saves the results of the module in the project database;
- based on the current design object descriptions in the property-oriented language of functional specifications generates a new description for the next stage of design.

In summary, it can be argued that the development of the mathematical bases, cognitive technologies, CAE, telecommunications, language theory and programming technologies made possible the design and implementation of the design-driven framework (DDF) network-centric Electronic Design Automation system.

In Fig. 1 a model of Design-Driven Framework (DDF) computing environment is presented. Here showing the blocks that provide the core functionality of DDF:

- *Object Description Parser* – unit, which provides analysis of functionally-oriented description of the design object, identifying the object for this narrative design phase, functional part of the design phase and quantitative description of the object and the application block;
- *Object Description Analyser* – unit designed for analysis of descriptions, select the available models, templates, IP blocks to form a complete model of design object, as well as for the analysis of functional part with a view to selecting the appropriate algorithms, modules, libraries to build code that implements the functionality of the design at this stage;

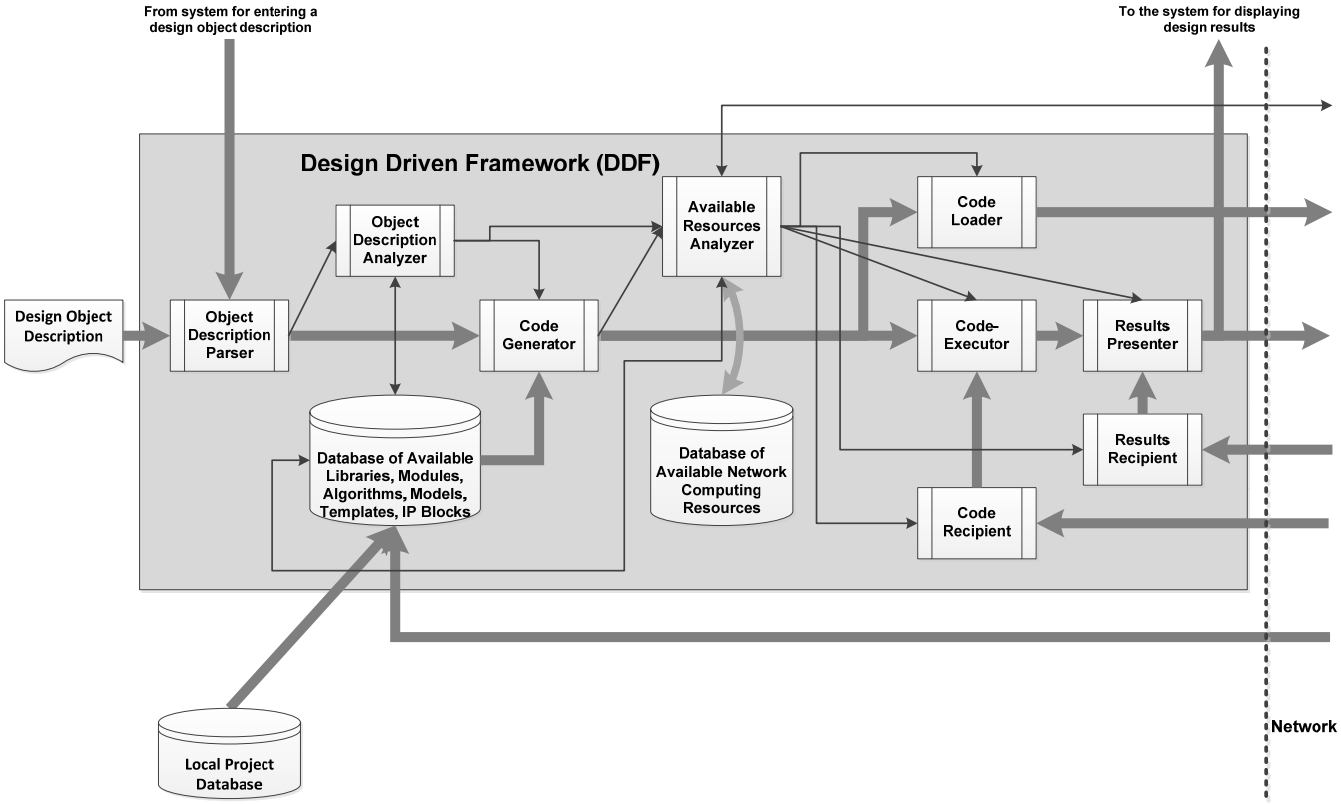

Fig.1. DDF model.

- *Code Generator* – is a module that provides a code that implements the functionality of the design at this stage of the currently available algorithms, modules, libraries (both locally and online);
- *Available Resources Analyser* – module that provides communication and exchange of management information between local DDF and all other DDF modules in the network. Is made up of two units:

o the *server part*, which has information on the current host, its current real time load, as well as on the composition of all the databases and libraries on local site that contain models, templates, IP blocks, algorithms, software modules, software libraries, which can be transmitted upon request to any host with the DDF computing environment;

o the *client part*, providing formation, sending and receiving network requests to search for nodes that can currently be prepared for execution of generated code that implements the required functionality for the design phase, as well as requests to the available network databases and libraries containing models, templates, IP blocks, algorithms, software modules, software libraries needed to generate such code.

- *Code Executor* – the code execution computing environment directly on the local node.
- *Code Loader* – block providing transfer of generated code to the node, selected by *Available Resources Analyser*, for remote execution there.
- *Code Recipient* – block to receive generated on the remote site code, when according to the list of available processing resources and current capacity, the local node was selected to execute this code.
- *Results Presenter* – block providing for post-processing the results of system operation in the current design stage for their visualization on the local node.
- *Results Recipient* – block providing for getting the results of the work of the DDF system at a remote site with a view to their subsequent transition to the local host.
- *Database of Available Network Computing Resources* – online database that stores the current state of all hosts and network resources available for use by the *Available Resources Analyser*.
- *Database of Available Libraries, Modules, Algorithms, Models, Templates, IP Blocks* – online database that stores references to all of these resources that are required to form a complete model of the design object and to build the code that implements the functionality of the design at this stage. It also stores references to libraries, modules, algorithms, models, templates, IP-blocks from the local database project. Is used by the *Object Description*

Analyser.

As you can see from Fig. 1, DDF communicates (or can communicate) with software modules and systems, located at higher level (layer) software:

- system for entering a description of the design object;
- system for displaying the results of the design and preparation of output files and documents of the design phase,

and at a lower level (layer):

- database management systems and the local project database;
- network protocols stack to enable communication over the network and merge all DDF modules into united virtual DDF computing environment.

Thus, we may speak of the Design-Driven Framework computing environment (DDF), how about program complex of middleware level, which is illustrated in Fig. 2.

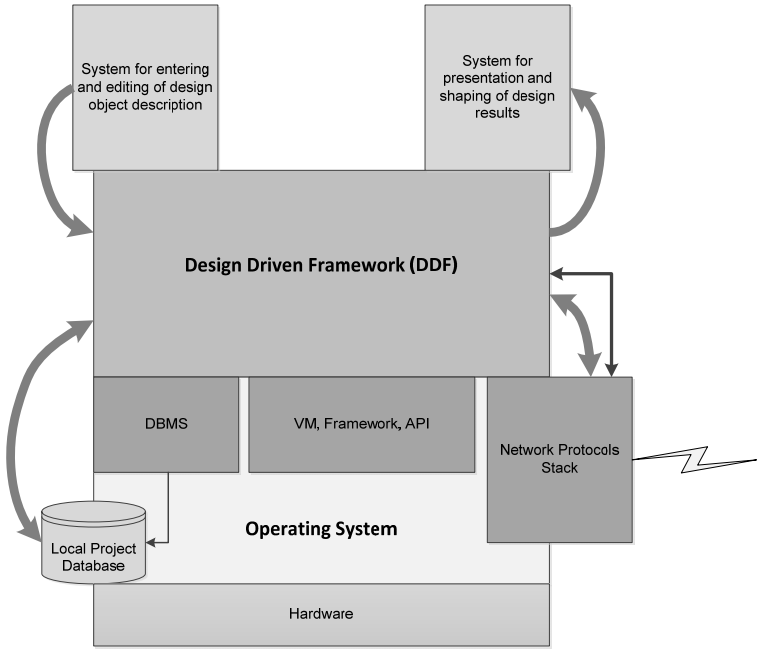

Fig.2. DDF as a middleware in a network node

Features of virtual computing environment based on Design-Driven Framework (DDF):

- Physical resources for DDF are allocated on request of availability. A specific node for functionality realization is not defined in advance.
- Restructuring of the logical structure of the network on the basis of the DDF occurs depending on events in the network and on the developing processes, including downloads, changes in the type and amount of the relationships between the nodes and the network environment
- It is allowed to quickly change behaviour rule sets for DDF.

Virtual clustering and metrics

We propose to introduce into consideration the *cluster infrastructure of the DDF computing environment of EDA systems*, characterized by the following features:

- Uses a distributed cluster infrastructure that provides immediate construction of virtual computing environment for each stage of end-to-end route of the design process in EDA systems;
- Applies a detailed discourse model of communication, focused on the interaction of all participants to work together to create high technology products;
- Implemented as a multi-agent system consisting of a group of interacting agents;
- Is the networking of active agents, each of which, depending on the situation and task, can act as a single managed entity, and in the role of unifying and coordinating (sending) a cluster, or the same as the upper level system (network), clusters, etc.
- The proposed infrastructure is grapho-variant system with network-centric resources [4].

The core element of the proposed infrastructure is a module that implements the concept of Design-Driven Framework (DDF) computing environment.

It is convenient to combine network computing resources in virtual clusters, using, for example, the metrics requirements to these resources from the project module is subject to execution in the network in a virtual DDF computing environment.

An example of paging of computer network with the DDF nodes on the virtual clusters is given in Fig. 3.

In this example, the network is divided into three clusters for the following metrics (parameters):

- A cluster can be used to handle modules with small memory requirements (maximum 4 GB) (cluster A),
- The amount of available memory in the site must be at least 16 GB (cluster B),
- The number of threads supported by the processor (CPU) of the site must be at least 16 (cluster C).

On the other hand, it is possible to combine other network resources (information, communication, cognitive) in their separate virtual clusters, using their own metrics (for example, membership of a specific resource to a specific stage of the end-to-end design route). Naturally, clustering for each type of network resources is completely independent with respect to another type of network resources.

Thus, we get the opportunity to implement a virtual cluster Design-Driven Framework (DDF) computing environment.

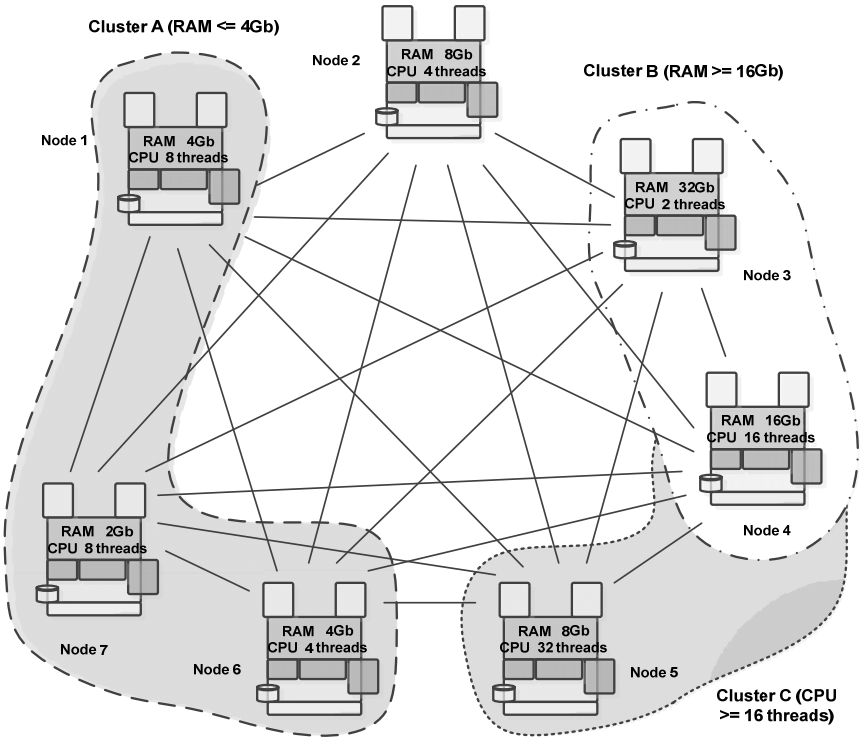

Fig.3. An example of virtual clustering

The implementation of EDA system for parametric optimization based on the method of *Abstract Bee Colony (ABC)* [5, 6] to simulate analogue and analogue-to-digital electronic circuits using virtual cluster DDF computing environment is one of the main activities currently being undertaken by the Computer-Aided Design Department in Saint-Petersburg State Electrotechnical University "LETI".

Summary

1. The interpenetration (convergence) of methods of computer-aided engineering and computational perception creates the basis for the emergence of a new kind of EDA – research Electronic Design Automation (rEDA).
2. We propose the implementation of the network-centric computing environments by creating in each node of network for each particular product (artifact) and moreover, for the various stages of its design, a separate middleware module, called Design-Driven Framework (DDF).
3. On the basis of the virtual DDF computing environment, it's possible to develop the methodology of language-oriented design (for example, the virtualization paradigm, conceptual design, connection to problem- and platform-oriented libraries, template-based design of hardware and software and so on), as well as to develop new strategies for EDA on the basis of network-centric architecture.
4. Proposed network-centric architecture provides an easy and convenient way to build a distributed DDF computing environment. Each node in the network can plan and implement the use of communication, computing, information and cognitive network resources needed to meet the challenges brought by this site. If the task to solve can be solved by means of parallel computing, the offered network-centric architecture provides the ability to implement parallel computing on all available nodes on the network that meet the relevant requirements.
5. Grapho-variation infrastructure of proposed network-centric architecture provides a virtual cluster computing environment in real time.
6. Proposed mechanism of virtual clustering provides a build of own computing environment for each stage of design for full end-to-end route design in EDA systems with the assistance of the network resources of various types and of different origin, located in different sites of available network space.

REFERENCES

[1] Osvalds G., Bowler M., Jones A. NCOIC Interoperability Framework // *NCOIC, 2008. URL: https://www.ncoic.org/technology/activities/education/nif/* (date of visit 07.10.2013)

[2] Base technologies of complementary design of high-tech products: Monograph / ed. I. V. Gerasimov and A. V. Nikitin, *ETU "LETI" Publishing house*, 2010, 196 pp. (in Russian)

[3] A. V. Nikitin, D. Ndayongeje, Property-oriented functional specification languages for conceptual stage of designing, *Trans. of ETU "LETI"*, 2010, No. 6, 25-30 (in Russian)

[4] A.Yu. Efremov, Yu.S. Zatuliveter, Yu.S. Legovich, Development of model and net-centric control algorithms for objects group locomotion using configuration of quasi-power fields, *Proc. of VI International Conf. "Parallel Computing and Control Problems" Moscow, Russia, October 24—26, 2012*, pp. 251-258 (in Russian)

[5] D. Karaboga, An Idea Based On Honey Bee Swarm For Numerical Optimization, *Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department,* 2005.

[6] Manish Gupta, Govind Sharma, An Efficient Modified Artificial Bee Colony Algorithm for Job Scheduling Problem, *IJSCE, Vol.1, Issue 6, Jan. 2012,* pp. 291-296

***Author**: Aleksandr Nikitin, candidate of technical sciences, associated professor of Computer-Aided Design Department in Saint-Petersburg State Electrotechnical University "LETI", ul. Professora Popova 5, 197376, Saint-Petersburg, Russia, E-mail: AVNike@gmail.com.*