Problem Solving Environment for Medical Image Analysis

Ketan C. Maheshwari[1], Sílvia D. Olabarriaga[1,2], Charl P. Botha[3],
Jeroen G. Snel[2], Johan Alkemade[2], Adam Belloum[1]
(1) Informatics Institute and (2) Academic Medical Center
University of Amsterdam, The Netherlands
(3) Data Visualization Group, Delft University of Technology, The Netherlands
{*kmaheshw,silvia,adam*}*@science.uva.nl*

Abstract

The development of Medical Image Analysis (MIA) applications that can successfully be applied in clinical practice is difficult for several reasons, one of them being the large amount and variety of resources involved (people, data, methods, computing). The application goes through several phases (development, parameter optimization, evaluation and clinical deployment) usually supported by different systems. The lack of support for information flow from phase to phase puts extra logistics burden on the lifecycle of MIA applications. In this paper we describe our first efforts to develop a Problem Solving Environment (PSE) for MIA applications using the three systems available at the proof-of-concept environment of the Virtual Laboratory for e-Sciences project. The proposed PSE implements data provenance mechanisms that support information flow among systems, facilitating navigation across phases of the application lifecycle.

1: Introduction

Medical Image Analysis (MIA) increasingly requires computational support to handle the size, heterogeneity, and complexity of the data and methods involved. The development of successful MIA applications that can be used in clinical care is a difficult process, requiring applications to go through several phases (*development*, parameter *optimization*, *evaluation* and clinical *deployment*). This process involves tasks that are performed by users playing different roles and using heterogeneous environments with complementary functionality. In practice, the systems operate in stand-alone fashion, and the users involved are not always aware of each other. In an ideal scenario, systems and users would interoperate and cooperate in an integrated computing environment [4].

This paper presents our first attempt to implement a Problem Solving Environment (PSE) to support the lifecycle of MIA applications as envisioned in [4]. Three systems are used to support the four main phases of the lifecycle, and the PSE provides mechanisms to facilitate their interoperability and information exchange across phases. This effort takes place in the scope of the Virtual Laboratory for e-Sciences Project (VL-e, *www.vl-e.com*), which aims at bridging the gap between developments in technology and application requirements by building PSE's for several domains. The paper is organized as follows: section 2 reviews the MIA application lifecycle as introduced in [4], briefly presenting the phases and the information flow among them. Section 3 presents the involved systems and the proposed PSE architecture, and in section 4 the first prototype is described. The paper concludes with a discussion of the initial results and plans for future work.

2: Lifecycle of MIA Applications

From development to deployment in the clinical routine, MIA applications go through phases in which different types of tasks are performed by professionals with varied backgrounds. Applications usually follow the component-based paradigm, in which image analysis functions are implemented by components that can be combined into processing networks. During *development* new applications (components or networks) are designed, developed and tested. The next phase is *optimization*, in which the optimal configuration of parameters is determined for a given class of situations (e.g. acquisition modality or a specific organ). During *evaluation* the MIA method is applied to large image collections in studies that determine their technical and clinical feasibility. Finally, in the *deployment* MIA applications are inserted in the clinical routine. Within the scope of MIA, software construction differs from the classical sequential waterfall model [5] in the following ways: (1) Each phase is performed in its own environment which is entirely alien to the environments belonging to other phases (2) each phase involves heterogeneous and distributed set of resources that span different departments or organizations.

Figure 1 provides an overview of the lifecycle of a MIA application, showing the information flow (data and applications) from phase to phase. Note that information also flows backwards in the lifecycle to feedback the process for application improvement. Different computational systems are adopted in each phase, and (usually) the information is manually transported from phase to phase in a process that is tedious and prone to errors. As proposed in [4] an integrated supporting environment would be valuable to facilitate navigation along the application's lifecycle, as well as to capture and enable reuse of data and methodology.

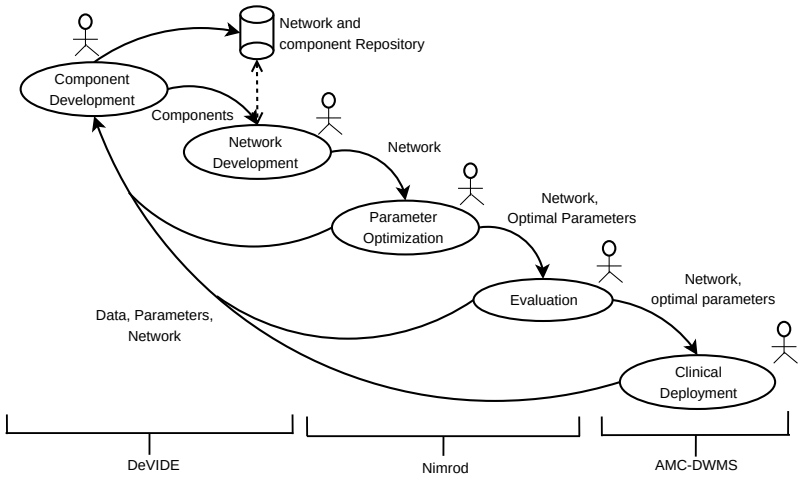

Figure 1. Information flow among phases of the MIA lifecycle, from development to clinical deployment, and the systems involved: DeVIDE, Nimrod, AMC-DWMS

3: PSE for MIA Applications

In the Medical Diagnosis and Imaging subprogram of the VL-e project, a PSE is under construction to facilitate the information flow illustrated in Figure 1. The VL-e "Proof-Of-Concept" environment (VL-e PoC) provides the physical infrastructure (storage, computing), services and systems shared by different application areas. The VL-e PoC resources are currently hosted by the SARA Computing and Networking Services and the Dutch Institute for Nuclear Physics and High Energy Physics (NIKHEF). Currently they consist of computational grid resources, a Storage

Resource Broker (SRB, *www.sdsc.edu/srb*), and a large number of services, including the systems adopted for MIA. The following systems (more details below) are used in this work: DeVIDE for the development, Nimrod for optimization (parameter sweeps) and evaluation (image sweeps), and the AMC-DWMS for clinical deployment. These systems offer complementary functionality necessary at different phases of a MIA application.

DeVIDE [2], or the Delft Visualization and Image processing Development Environment, is a platform-independent, interactive system that facilitates the rapid implementation and testing of component-based applications. DeVIDE acts as an application development environment. DeVIDE integrates extensive visualization and image processing functionality from libraries like VTK (*www.vtk.org*) and ITK (*www.itk.org*) and provides a visual programming language to compose processing networks by connecting high-level functional components. DeVIDE has two main modes of operation: in interactive mode, it facilitates run-time steering of network execution via a rich graphical user interface from which parameter values and module properties can be experimented with. In command-line mode, saved networks can be restored and configured to be executed without the GUI. In this way, complete networks can be visually constructed and explored, and later used as black-box third-party components in other applications.

Nimrod [1] facilitates the management of large experiments on distributed resources by providing parameter sweep (Nimrod-G) and optimization (Nimrod/O) functionalities. The experiment is presented to Nimrod in a simple "plan" describing the files to be transferred to the execution node (stage-in), the tasks to be executed, and the results to be transferred back to the source node (stage-out). Nimrod dynamically discovers, acquires and monitors computational resources and adaptively schedules the tasks among them. It keeps the experiment status (running tasks, retries, etc.) in a database that can be queried by the user.

The **AMC-DWMS** [6], or the Academic Medical Center Distributed Workflow Management System, performs the deployment of custom MIA applications in a desktop grid inside the hospital. AMC-DWMS acts as a clinical image analysis system that automatically performs data logistics and processing based on pre-defined workflows programmed in XML. The system supports a large variety of tasks, such as image import-export, caching, analysis and notification services. A relational database system is used to store and retrieve the information related to workflow actions. An interface called 'control-center' is used to install, configure, update, synchronize and monitor activities. The system supports the DICOM (Digital Image and COMmunication) protocol to exchange image data between acquisition systems (MRI, CT-scanners), image archives (PACS) and display workstations, seamlessly integrating the MIA applications into the radiology environment of a hospital.

Architecture. The proposed architecture is illustrated in Figure 2. It supports the information flow among different application lifecycle phases via data provenance mechanisms [3]. Three layers (User Interface, Web-Services and Database) along with legacy systems (AMC-DWMS, Nimrod and DeVIDE) form a framework for managing and organizing the information associated with a MIA application. The web-services layer consists of Application Description, Data Provenance, Project and Application Execution management services. These services are invoked through stubs extended from the legacy systems. The Virtual File System (VFS) layer abstracts the database. *Application Description* services provide a mechanism to describe, store, access and retrieve MIA applications and their versions. *Application Execution* services are used to store information that needs to be tracked from phase to phase, including applications, parameters, image data, and standard logs. A project is an aggregation of the information associated with a single MIA problem, containing sessions that correspond roughly to phases in the MIA lifecycle. Users can subscribe to the events in a project. *Project management* services facilitate the creation, access, subscription and cleaning up of projects. The user interface layer provides user-friendly interaction with the system. Using the *Data Provenance* services, the user can retrieve provenance data to navigate back during the MIA lifecycle. The complete settings used to generate a given result can be restored and used to run the network interactively in DeVIDE for debugging purposes.

Applications are developed using DeVIDE interactively and, after sufficient testing, the network

is saved in the *Application Registry* using the Application Description services. Nimrod is used to run large experiments that invoke the application for parameter sweeps (optimization) or image sweeps (evaluation). Finally, the application is inserted into a AMC-DWMS systems, being executed automatically after image acquisition. Information related to each instance of application executoion is stored in the *Provenance Store* using the Application Execution services.

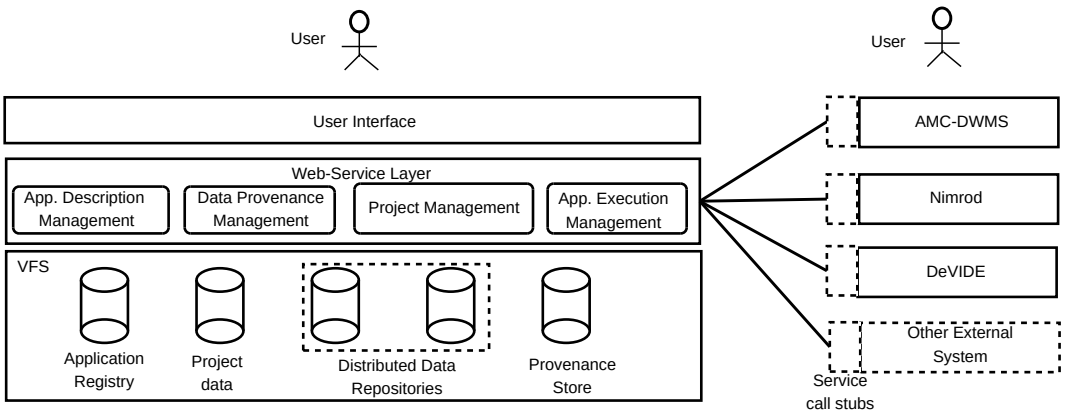

Figure 2. Proposed Architecture for the PSE

4: Prototype

An experimental setup was implemented to realize all phases of a simple MIA application using the proposed PSE. Both Nimrod and AMC-DWMS run DeVIDE networks in command-line mode. Before running the DeVIDE application, the data and DeVIDE network are staged in according to the Nimrod plan or the AMC-DWMS workflow. When the application is completed, the output results are staged back. Currently no real data provenance mechanisms are implemented; instead, all data (input/output images), standard log files (stdout,stderr) and DeVIDE networks are stored in the local file system. At a later stage, the user can manually retrieve data from the saved files and restore the interactive DeVIDE application with the same settings.

Example Application. A region-of-interest (ROI) is segmented from an image using a threshold operation. A DeVIDE network imports the image and a threshold range (TR) as parameters, and outputs a binary mask for the voxels within the chosen TR. Different ROIs can be selected by choosing a proper (optimal) TR. Here we focus on segmentation of blood vessels from contrast-enhanced CT Angiography scans of the head.

Development. A DeVIDE network was developed to implement the segmentation application using available components in DeVIDE's library. The network contains a VTK image reader, a 'DoubleThreshold' image processing operator, a custom 'Slice3DViewer' and a VTK image writer. During development, the user runs DeVIDE interactively, setting the TR on the GUI and observing the generated result in a viewer (see Figure 3).

Optimization/Evaluation. Nimrod was used to investigate the optimal TR to segment blood vessels. A parameter sweep on the upper threshold value was performed, while keeping the lower threshold fixed (1200 HU). Figure 4-left shows the Nimrod plan for varying the threshold between 1200 HU and 1600 HU. The data and executables are staged-in, DeVIDE is executed in command-line mode, providing the image, parameters and the network as arguments, and the output data is staged-out. The resulting images are visually inspected by an expert to determine the optimal threshold value to segment blood vessels (i.e. 1210 HU). In a similar fashion, an image sweep was performed with Nimrod to run the method with the optimal settings for many images.

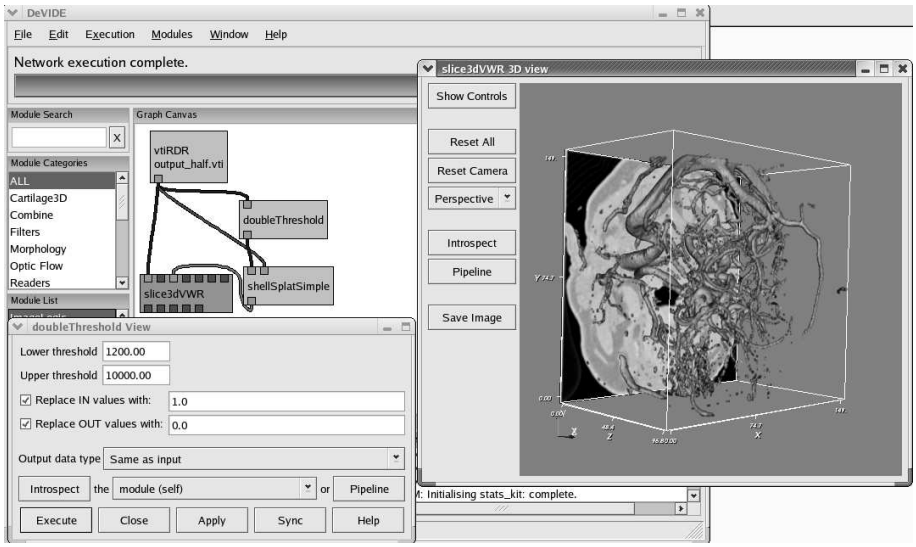

Figure 3. DeVIDE network in interactive mode during the development phase.

Clinical Deployment. AMC-DWMS was used to insert this application into a simulated clinical setting. The complete workflow consists of the following tasks (see Figure 4-right): import the CT scan (DICOM) data from a server, convert it to the appropriate VTK data format, start the DeVIDE application with the optimal parameters, convert output from VTK to DICOM format, and export results to a DICOM node.

Feedback in Lifecycle. During the inspection of results produced during optimization, evaluation or deployment, a medical or technical expert can detect problems (e.g., degraded output, algorithmic instability). Using the information saved in the log files, it is possible to reload the DeVIDE network in interactive mode with the input image and parameters that generated the problem. The user can then add more components to the network (e.g. other 3D viewers) to comfortably investigate the situation at hand.

5: Discussion and Future Work

We proposed a general architecture of a PSE to facilitate the information flow among systems supporting different phases of a MIA application lifecycle in the VL-e project. The PSE is composed of existing systems and a mechanism to track provenance data that can be used to navigate across phases. Information is saved to enable invoking the application in DeVIDE interactive mode at a later stage, when errors are detected during optimization, evaluation or deployment.

Several practical challenges had to be faced initially. Currently all systems work on compatible platforms, and a simple logging mechanism enables information flow among them. With the prototype and simple application presented in this paper, we have demonstrated that it is possible to construct a PSE that assists across all phases of MIA development for clinical applications. A proposed algorithm can be interactively developed in a rich graphical environment, then seamlessly transported through parameter optimization and evaluation phases, and finally deployed in clinical practice, all using a unified set of tools. At any stage return to a previous phase in the lifecycle is possible, with the prospect of facilitating the maintenance of MIA applications that meet the standards of health care.

The implementation described here shows a minimal setup in which the systems were adapted to run in the same platform following the requirements of the VL-e PoC. Using this experimental set-up, we now focus on the supporting data provenance mechanisms, which are essential for the

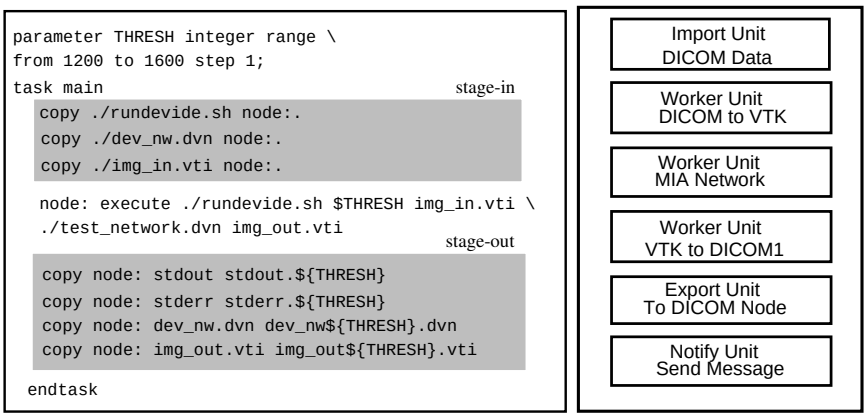

Figure 4. Nimrod plan for parameter optimization and AMC-DWMS Workflow

design and implementation for the proposed PSE. Our future work concentrates on the investigation of extensive provenance requirements and implementation of provenance mechanism using standard practices.

Acknowledgments

We thank our VL-e colleagues, R. Belleman, B. Ó. Nualláin, and S. Marshall for their contribution to this project. We are thankful to Prof. Dr. L.O. Hertzberger (VL-e director) and the VL-e coordinators at the AMC, Prof. Dr. C.A. Grimbergen and Prof. Dr. G.J. den Heeten for their support. This work was carried out in the context of the VL-e project, which is supported by a BSIK grant from the Dutch Ministry of Education, Culture and Science (OC&W) and is part of the ICT innovation program of the Ministry of Economic Affairs (EZ).

References

[1] D. Abramson et al. Nimrod: A tool for performing parametised simulations using distributed workstations. In *The 4th IEEE Symposium on High Performance Distributed Computing*, 1995.

[2] C.P. Botha. DeVIDE - The Delft Visualization and Image Processing Development Environment. Technical report, TU Delft, May 2005. http://cpbotha.net/DeVIDE.

[3] Peter Buneman, Sanjeev Khanna, and Wang-Chiew Tan. Data provenance: Some basic issues. In *Foundations of Software Technology and Theoretical Computer Science*, 2000.

[4] S. Olabarriaga, J.G. Snel, C.P. Botha, and R.G. Belleman. Integrated support for medical image analysis methods: from development to clinical application. *IEEE Transactions on Information Technology in Biomedicine, January 2007, Volume: 11, Issue: 1, page(s): 47-57.*

[5] R.S. Pressman. *Software Engineering: A Practitioner's Approach.* McGraw Hill, 2000.

[6] J. G. Snel et al. A Distributed Workflow Management System for Automated Medical Image Analysis and Logistics. In *19th IEEE Symposium on Computer-Based Medical Systems (CBMS'06) special track on Grids for Biomedical Informatics*, 2006.