Multi-layer Architecture For Storing Visual Data Based on WCF and Microsoft SQL Server Database

Rafał Grycuk[1], Marcin Gabryel[1], Rafał Scherer[1], Sviatoslav Voloshynovskiy[2]

[1] Institute of Computational Intelligence, Częstochowa University of Technology
Al. Armii Krajowej 36, 42-200 Częstochowa, Poland
{rafal.grycuk, marcin.gabryel, rafal.scherer}@iisi.pcz.pl
http://iisi.pcz.pl
[2] University of Geneva, Computer Science Department,
7 Route de Drize, Geneva, Switzerland
group web page: http://sip.unige.ch

Abstract. In this paper we present a novel architecture for storing visual data. Effective storing, browsing and searching collections of images is one of the most important challenges of computer science. The design of architecture for storing such data requires a set of tools and frameworks such as SQL database management systems and service-oriented frameworks. The proposed solution is based on a multi-layer architecture, which allows to replace any component without recompilation of other components. The approach contains five components, i.e. Model, Base Engine, Concrete Engine, CBIR service and Presentation. They were based on two well-known design patterns: Dependency Injection and Inverse of Control. For experimental purposes we implemented the SURF local interest point detector as a feature extractor and K-means clustering as indexer. The presented architecture is intended for content-based retrieval systems simulation purposes as well as for real-world CBIR tasks.

Keywords: WCF, Microsoft SQL Server, Dependency Injection, Inversion of Control, Entity Framework, Multi-layer Architecture, k-means, SURF, Content-Based Image Retrieval

1 Introduction

Images are created everyday in tremendous amount and there is ongoing research to make it possible to efficiently search these vast collections by their content. Generally, this work can be divided into image classification [4][18] and image retrieval [17]. Recognizing images and objects on images relies on suitable feature extraction which can be basically divided into several groups, i.e. based on color representation [28], textures [48], shape [50][52], edge detectors [42][43][53] or local invariant features, e.g. SURF [2], SIFT [34] or ORB [46]. Matching features

can be also performed by several methods, e.g. clustering, nearest neighbour, bag of features [21] or soft computing [15][30].

There are many content-based image processing systems developed so far. A good review of such systems is provided in [51]. To the best of our knowledge no other system uses similar set of tools to the system proposed in the paper. Now we describe briefly the most important tools used to design the proposed framework.

1.1 Windows Communication Foundation

Windows Communication Foundation (WCF) is a framework based on Service-Oriented Architecture [14][27]. WCF allows to send data asynchronously between two service and client endpoints. Service endpoints can be deployed on IIS server or be hosted locally. The messages can be send as XML (value types) or binary file (complex types) [9][29][35][39]. WCF consist of following features:[45][49] Service Orientation, Interoperability, Multiple Message Patterns, Service Metadata, Data Contracts, Security, Multiple Transports and Encodings, Reliable and Queued Messages, Durable Messages, Transactions, AJAX and REST Support, Extensibility.

1.2 SQL Sever

MS SQL Server is a database management system for storing various types of data, fully supporting cloud computing technologies [5][11][12][16]. It provides a set of tools to extract data from various devices or sources, even at datacenters. MS SQL query language, T-SQL (Transact-SQL), allows for both structural or procedural queries [6][10][13][31][37][38]. The DBMS (Database Management System) is based on a client-server architecture. The platform is composed of the following services:[33][41]

- Database engine - allows to execute queries and is necessary to run the server,
- Integration Services (SSIS, SQL Server Integration Services) - ETL (Extraction, Transformation and Loading) platform responsible for data migration from the heterogeneous data sources,
- SQL Agent - answerable for performing tasks according to the specified schedule,
- Full-text Filter Daemon Launcher - allow to perform full-text searches on text columns,
- Reporting Services (SSIS, SQL Server Reporting Services) - responsible for designing and deploying reports,
- Analysis Services (SSAS, SQL Server Analysis Services) - allows to create multidimensional cubes and executing MDX (Multidimensional Expressions) queries [25],

DBMS store data in a relational form (tables and their relations) and allows to select information by executing queries. Many frameworks perform object relational mapping. In this paper we use Entity Framework (EF) with Code First approach [1][8][32].

1.3 Dependency Injection and Inverse of Control

Re-usability of existing components is crucial in modern software engineering. The aim of this approach is to combine separate layers into one application. This is a challenging task, because as the application complexity increases, so do dependencies [7]. The best practice for tone down proliferation of dependencies is by using Dependency Injection (DI) design pattern allowing to inject objects into a class constructor. Thus, the creation of the object does not rely on a class. The initialization logic is rarely reusable outside of created component. That pattern provides a layer of abstraction for the injected object, thus we can implement the concrete logic in the other component and inject it in the class constructor by the interface. DI [39] is an implementation of Inverse of Control (IoC)[20][44][47]. Figure 1 shows the typical class dependencies. Such a scheme entails the following problems [36]:

1. Any code changes of *ServiceA* forces changes in *ClassA*, thus the recompilation of all components is required,
2. All classes must be implemented and available at the compile time,
3. Classes are difficult to test and to achieve components isolation,
4. Contradicts the (DRY) Don't Repeat Yourself principle.

To resolve this issue we used the Dependency Injection [7][20]. A Conceptual view of DI is presented in Figure 2.

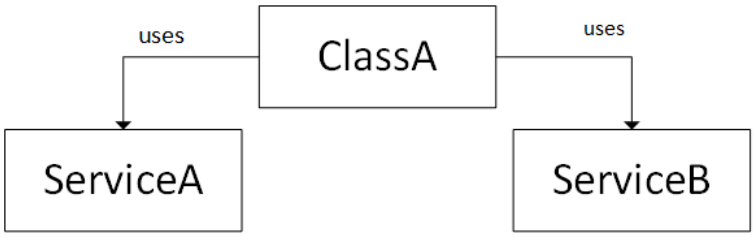


Fig. 1. Problem illustration of tightly coupled dependencies. *ClassA* uses *ServiceA* and *ServiceB*. This is a simple representation of *ClassA* dependencies on *ServiceA* and *ServiceB*.

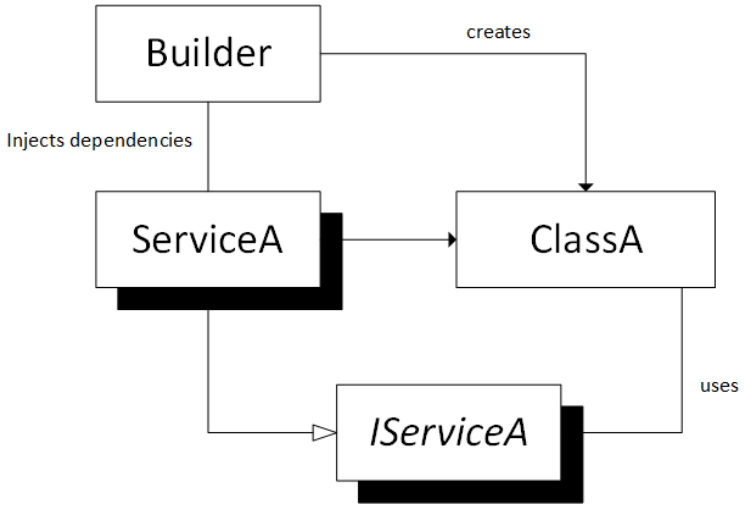


Fig. 2. Solution of the problem formulated in Fig. 1. The Inverse of Control pattern can be implemented by Dependency Injection. *Builder* creates *ClassA* which uses abstract interface *IServiceA*. Dependencies of *ServiceA* are injected to *ClassA* by inheritance.

2 Proposed Architecture For Storing Visual Data

Proposed architecture is based on two main components. The first one is SQL database which is Microsoft SQL Server, and the second one is Windows Communication Foundation (WCF). Our approach consists of five main layers (tiers):

1. Model - which contains data model generated by the Entity Framework (6.1) with Code First approach,
2. Base Engine - consist of several abstract classes or interfaces which can be used to implement user solutions, but they provide appropriate business logic,
3. Concrete Engine - implements user logic based on previous the layer (in this paper we implemented the SURF descriptors for feature extraction and k-means clustering [26] for indexing),
4. CBIR Service (for more about CBIR see [23][22]) - which is WCF service that allows to invoke engine methods as Service Oriented Applications (SOA).
5. Endpoint (Client) - presentation layer for invoking service methods, it can be desktop, web or mobile application. All the user needs to do, is to add service reference and invoke methods.

The agility applied in the presented approach is important, because it is not restricted to any particular implementation and it can be applied in various solutions. The architecture is presented in Fig. 3a.

The Base Engine and Concrete Engine layers are based on two design patterns, i.e. Dependency Injection (DI) and Inverse of Control (IoC) described

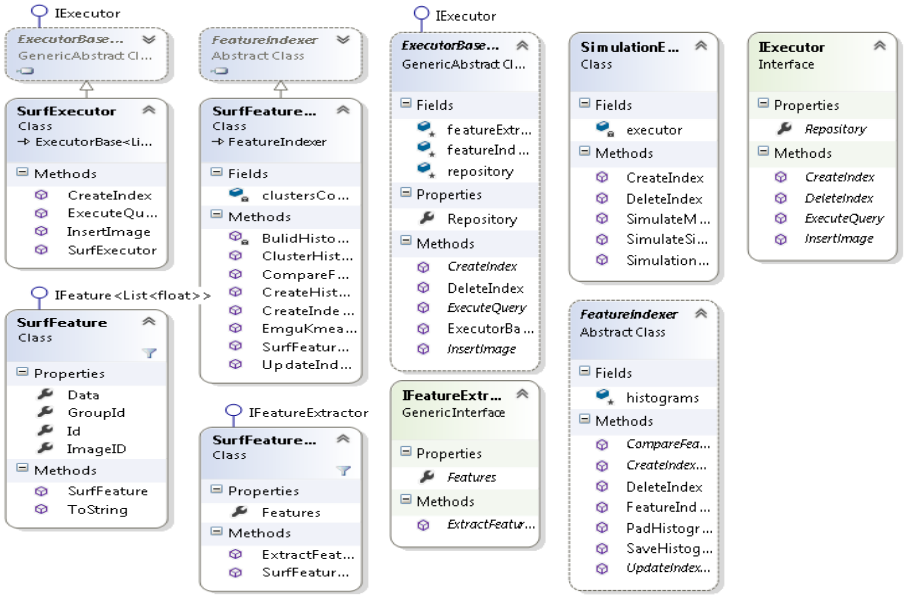
(a) Multi-tier architecture for storing visual data. Final components are compiled release of dll's and open components are user implementation.

(b) Entity diagram based on database tables.

Fig. 3. Multi-layer architecture and entity diagram of the proposed approach.

in Section 1.3. These patterns allowed to separate the containers and maintain the S.O.L.I.D. principles [39]. Components are composed of a set of classes or interfaces. Each component has a representation in UML (Unified Modeling Language) diagram. Figure 5a shows the model layer, which contains five classes and two interfaces. Classes: *Bin*, *Histogram* and *Image* were generated by the Entity Framework and they correspond with the database tables presented in Fig. 3b. Interface *IRepositoryBase* is a generic interface which provides the basic C.R.U.D. (create, read, update and delete) operations. The methods allow to operate on any types of objects. The *GenericRepository* implements *IRepositoryBase* interface. In addition, the *dataContext* field is generic, thus the concrete implementation does not contains any dependencies. The *GenericRepository* class is based on the Singleton pattern, to create instance which user needs to use static method *GetInstance*. A very interesting interface is *IFeature*, which allows to implement any type of image features (an image descriptor).

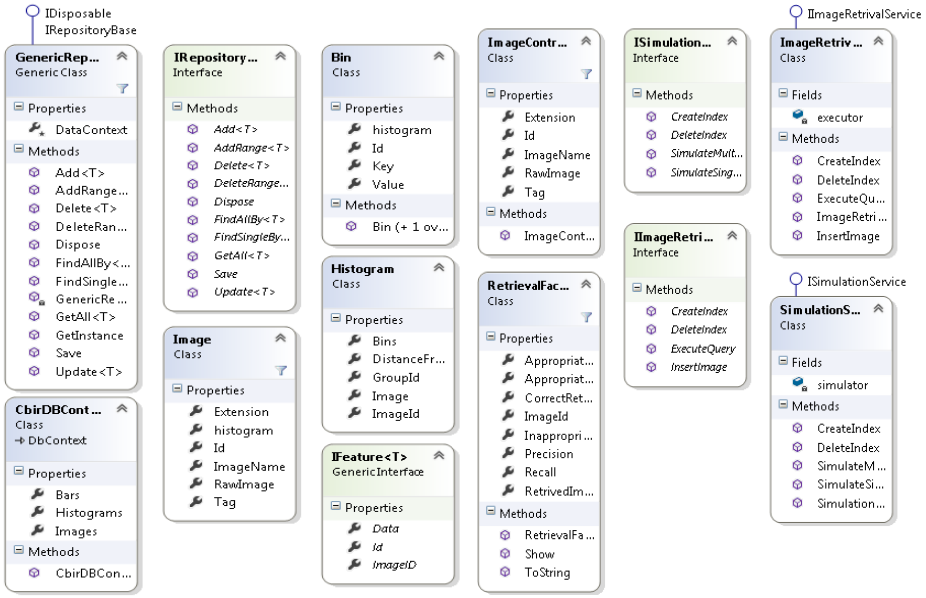
Fig. 4b represents a class diagram for the Base Engine layer, which contains items for feature extraction (*IFeatureExtractor*), feature indexation (*Feature Indexer*), operations executions (*IExecutor*, *ExecutorBase*) and simulation (*SimulationEvaluator*, *RetrievalFactors*). *IExecutor* provides abstract methods for following operations: index creation and deletion, query execution and image insertion. It contains one property: repository. *IFeatureExtractor* is re-



(a) Class diagram for Concrete Engine layer.

(b) Class diagram for Base Engine layer.

Fig. 4. Class diagrams for Concrete Engine and Base Engine layers.



(a) Class diagram for Model layer. (b) Class diagram for CBIR Service layer.

Fig. 5. Class diagrams for Model and CBIR Service layers.

sponsible for feature extraction and contains: *Features* property and ExtractFeatures method. The first one stores previously extracted features, the second one extracts features from image passed as a parameter. The method is only a definition, thus the class that will implement that interface must contain its own version. *FeatureIndexer* consists of one field (histograms) which is a composition relation with the *Histogram* class. Methods are abstract, thus they require concrete implementation in the inherited class. *ExecutorBase* is a base class that uses the dependency injection to initiate object, that allows to inject logic. The *RetrievalFactors* contains fields that describes the query results. The Concrete Engine layer (see Fig. 4a) is designed and presented for simulation purposes. We implemented SURF [3][19][24] as a feature extractor and k-means method as a indexer. Figure 5b shows diagram for the CBIR Service layer. Each WCF service consists of the following items:

- Interface - defines the method e.g. *IImage Retrival Service*,
- Implementation class - implements the method's body e.g. *Image Retrieval Service*,
- Contract (optional) - required to retrieve data from the service *Image Contract*.

CBIR Service component contains two services: simulation service is used to perform simulations on the created index. The most interesting methods are SimulateMultiQuery and SimulateSingleQuery. The first one returns a list of

RetrievalFactors and performs multi query. The second one executes a single query. The image retrieval service allows to execute queries. The difference between SimulateSingleQuery and ExecuteQuery methods are the following: ExecuteQuery returns a list of retrieved images, SimulateSingleQuery returns a list of factors (*precision*, *recall*) which allows to evaluate index efficiency. The proposed architecture was designed in .NET framework and written in C#.

3 Experimental Results

Experiments were carried out using the designed architecture. The created endpoint was a desktop application with service reference to CBIR Service. Research includes experiments on various objects with background described by the SURF local interest point descriptor [34]. Test images were taken from the Corel database. We chose images with various types of objects. In experiments we used 90% of each class for index creating and 10% as query images. In Tab.1 we presented the retrieved factors for each query image. Tab. 2 shows retrieved images for a query image (the image with border).

For the purposes of the performance evaluation we use two measures; precision and recall [40]. Fig. 6 shows the performance measures of the image retrieval. The AI is a set of appropriate images, that should be returned as being similar to the query image. The RI represents a set of returned images by the system. Rai is a group of properly returned images. Iri represents improperly returned images, anr proper not returned and inr improper not returned images. The presented measure allows to define *precision* and *recall* by the following formulas [40]:

$$precision = \frac{|rai|}{|rai + iri|}, \tag{1}$$

$$recall = \frac{|rai|}{|rai + anr|}. \tag{2}$$

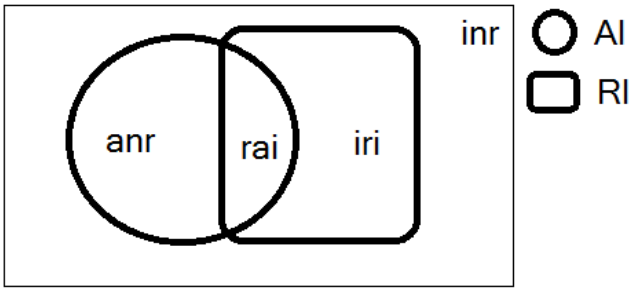


Fig. 6. Performance measures diagram.

Image Id	RI	AI	rai	iri	anr	Precision	Recall
1 (1).jpg	19	33	21	12	7	13	7
1 (2).jpg	21	33	31	3	18	3	19
1 (20).jpg	19	33	21	12	7	13	7
1 (21).jpg	19	33	21	12	7	13	7
2 (1).jpg	14	33	27	7	7	7	8
2 (10).jpg	12	33	28	6	6	6	7
2 (11).jpg	19	33	24	9	9	10	10
2 (17).jpg	12	33	28	6	6	6	7
3 (1).jpg	22	33	23	11	12	11	12
3 (10).jpg	21	33	23	10	11	11	12
3 (11).jpg	18	33	27	7	11	7	12
3 (15).jpg	21	33	29	4	17	4	17

Table 1. Simulation results for multi query. Measures were normalized and presented as percentage value [%].

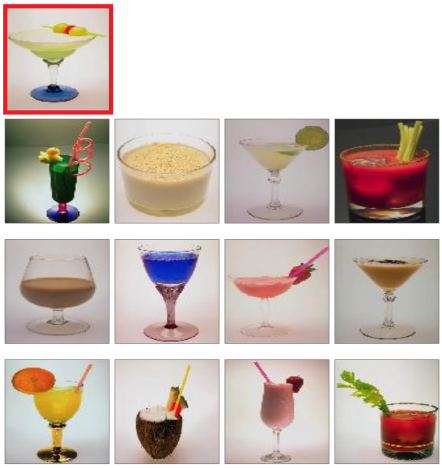


Table 2. Query results. Eighteen example images from the experiment. The image with border is the query image.

4 Final Remarks

The presented approach is a novel architecture for storing visual data. We used SQL Server and WCF services as a core of our method. The proposed solution for storing visual data has no dependencies with concrete implementation, thus we can simulate any CBIR method. Our approach allows creating any type of endpoint: desktop, web or mobile application. It can be used as a core backend solution. The performed experiments proved effectiveness of our method. Our paper present a part of a larger system that allows to search and identify specific classes of objects.

Acknowledgments

The work presented in this paper was supported by a grant BS/MN-1-109-301/14/P "Clustering algorithms for data stream - in reference to the Content-Based Image Retrieval methods (CBIR)". The work presented in this paper was supported partially by a grant from Switzerland through the Swiss Contribution to the enlarged European Union.

References

[1] Adya, A., Blakeley, J.A., Melnik, S., Muralidhar, S.: Anatomy of the ado. net entity framework. In: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, ACM (2007) 877–888

[2] Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (surf). Comput. Vis. Image Underst. **110**(3) (June 2008) 346–359
[3] Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In: Computer Vision–ECCV 2006. Springer (2006) 404–417
[4] Bazarganigilani, M.: Optimized image feature selection using pairwise classifiers. Journal of Artificial Intelligence and Soft Computing Research **1**(2) (2011) 147–153
[5] Ben-Gan, I.: Microsoft SQL Server 2012 T-SQL Fundamentals. Pearson Education (2012)
[6] Biniaz, A., Abbasi, A.: Segmentation and edge detection based on modified ant colony optimization for iris image processing. Journal of Artificial Intelligence and Soft Computing Research **3**(2) (2013) 133–141
[7] Caprio, G.: Design patterns-dependency injection. MSDN Magazine (2005) 103–110
[8] Castro, P., Melnik, S., Adya, A.: Ado. net entity framework: raising the level of abstraction in data programming. In: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, ACM (2007) 1070–1072
[9] Chappell, D.: Introducing windows communication foundation in .net framework 4. Retrieved May **11** (2010) 2011
[10] Chen, M., Ludwig, S.A.: Particle swarm optimization based fuzzy clustering approach to identify optimal number of clusters. Journal of Artificial Intelligence and Soft Computing Research **4**(1) (2014) 43–56
[11] Chromiak, M., Stencel, K.: The linkup data structure for heterogeneous data integration platform. In: Future Generation Information Technology - 4th International Conference, FGIT 2012, Gangneug, Korea, December 16-19, 2012. Proceedings. (2012) 263–274
[12] Chromiak, M., Stencel, K.: A data model for heterogeneous data integration architecture. In: Beyond Databases, Architectures, and Structures - 10th International Conference, BDAS 2014, Ustron, Poland, May 27-30, 2014. Proceedings. (2014) 547–556
[13] Chu, J.L., Krzyźak, A.: The recognition of partially occluded objects with support vector machines, convolutional neural networks and deep belief networks. Journal of Artificial Intelligence and Soft Computing Research **4**(1) (2014) 5–19
[14] Cibraro, P., Claeys, K., Cozzolino, F., Grabner, J.: Professional WCF 4: Windows Communication Foundation with. NET 4. John Wiley & Sons (2010)
[15] Cpalka, K.: On evolutionary designing and learning of flexible neuro-fuzzy structures for nonlinear classification. Nonlinear Analysis: Theory, Methods & Applications **71**(12) (2009) e1659 – e1672
[16] Delaney, K., Freeman, C.: Microsoft SQL Server 2012 Internals. Microsoft Press (2013)
[17] Drozda, P., Sopyla, K., Górecki, P.: Online crowdsource system supporting ground truth datasets creation. In: Artificial Intelligence and Soft Computing - 12th International Conference, ICAISC 2013, Zakopane, Poland, June 9-13, 2013, Proceedings, Part I. (2013) 532–539
[18] Drozda, P., Sopyla, K., Górecki, P.: Different orderings and visual sequence alignment algorithms for image classification. In: Artificial Intelligence and Soft Computing - 13th International Conference, ICAISC 2014, Zakopane, Poland, June 1-5, 2014, Proceedings, Part I. (2014) 693–702
[19] Evans, C.: Notes on the opensurf library. University of Bristol, Tech. Rep. CSTR-09-001, January (2009)

[20] Fowler, M.: Inversion of control containers and the dependency injection pattern (2004)
[21] Grauman, K., Darrell, T.: Efficient image matching with distributions of local invariant features. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Volume 2. (June 2005) 627–634 vol. 2
[22] Grycuk, R., Gabryel, M., Korytkowski, M., Romanowski, J., Scherer, R.: Improved digital image segmentation based on stereo vision and mean shift algorithm. In Wyrzykowski, R., Dongarra, J., Karczewski, K., Waniewski, J., eds.: Parallel Processing and Applied Mathematics. Volume 8384 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2014) 433–443
[23] Grycuk, R., Gabryel, M., Korytkowski, M., Scherer, R.: Content-based image indexing by data clustering and inverse document frequency. In Kozielski, S., Mrozek, D., Kasprowski, P., Maysiak-Mrozek, B., Kostrzewa, D., eds.: Beyond Databases, Architectures, and Structures. Volume 424 of Communications in Computer and Information Science. Springer International Publishing (2014) 374–383
[24] Grycuk, R., Gabryel, M., Korytkowski, M., Scherer, R., Voloshynovskiy, S.: From single image to list of objects based on edge and blob detection. In Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L., Zurada, J., eds.: Artificial Intelligence and Soft Computing. Volume 8468 of Lecture Notes in Computer Science. Springer International Publishing (2014) 605–615
[25] Harinath, S., Pihlgren, R., Lee, D.G.Y., Sirmon, J., Bruckner, R.M.: Professional Microsoft SQL Server 2012 Analysis Services with MDX and DAX. John Wiley & Sons (2012)
[26] Hartigan, J.A., Wong, M.A.: Algorithm as 136: A k-means clustering algorithm. Applied statistics (1979) 100–108
[27] Hirschheim, R., Welke, R., Schwarz, A.: Service-oriented architecture: Myths, realities, and a maturity model. MIS Quarterly Executive **9**(1) (2010) 37–48
[28] Huang, J., Kumar, S., Mitra, M., Zhu, W.J., Zabih, R.: Image indexing using color correlograms. In: Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on. (Jun 1997) 762–768
[29] Klein, S.: Professional WCF programming:. NET development with the Windows communication foundation. John Wiley & Sons (2007)
[30] Łapa, K., Zalasiski, M., Cpałka, K.: A new method for designing and complexity reduction of neuro-fuzzy systems for nonlinear modelling. In Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L., Zurada, J., eds.: Artificial Intelligence and Soft Computing. Volume 7894 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 329–344
[31] Leblanc, P.: Microsoft SQL Server 2012 Step by Step. Pearson Education (2013)
[32] Lerman, J., Miller, R.: Programming Entity Framework: Code First. " O'Reilly Media, Inc." (2011)
[33] Lobel, L., Brust, A.: Programming Microsoft® SQL Server® 2012. " O'Reilly Media, Inc." (2012)
[34] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision **60**(2) (November 2004) 91–110
[35] Mackey, A.: Windows communication foundation. In: Introducing. NET 4.0. Springer (2010) 159–173
[36] Magazine, M.: Design patterns: Dependency injection (2005)

[37] Makinana, S., Malumedzha, T., Nelwamondo, F.V.: Quality parameter assessment on iris images. Journal of Artificial Intelligence and Soft Computing Research **4**(1) (2014) 21–30
[38] Mallik, S., Mukhopadhyay, A., Maulik, U.: Integrated statistical and rule-mining techniques for dna methylation and gene expression data analysis. Journal of Artificial Intelligence and Soft Computing Research **3**(2) (2013) 101–115
[39] Martin, M., Martin, R.C.: Agile principles, patterns, and practices in C#. Pearson Education (2006)
[40] Meskaldji, K., Boucherkha, S., Chikhi, S.: Color quantization and its impact on color histogram based image retrieval. (2009)
[41] Mistry, R., Misner, S.: Introducing Microsoft® SQL Server® 2012. " O'Reilly Media, Inc." (2012)
[42] Ogiela, M.R., Tadeusiewicz, R.: Syntactic reasoning and pattern recognition for analysis of coronary artery images. Artificial Intelligence in Medicine **26**(1) (2002) 145–159
[43] Ogiela, M.R., Tadeusiewicz, R.: Nonlinear processing and semantic content analysis in medical imaging-a cognitive approach. Instrumentation and Measurement, IEEE Transactions on **54**(6) (2005) 2149–2155
[44] Prasanna, D.R.: Dependency injection. Manning Publications Co. (2009)
[45] Rosen, M., Lublinsky, B., Smith, K.T., Balcer, M.J.: Applied SOA: service-oriented architecture and design strategies. John Wiley & Sons (2012)
[46] Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: Orb: An efficient alternative to sift or surf. In: Computer Vision (ICCV), 2011 IEEE International Conference on. (Nov 2011) 2564–2571
[47] Seemann, M.: Dependency injection in. NET. Manning (2012)
[48] Śmietański, J., Tadeusiewicz, R., Łuczyńska, E.: Texture analysis in perfusion images of prostate cancera case study. International Journal of Applied Mathematics and Computer Science **20**(1) (2010) 149–156
[49] Tsai, W.T., Sun, X., Balasooriya, J.: Service-oriented cloud computing architecture. In: Information Technology: New Generations (ITNG), 2010 Seventh International Conference on, IEEE (2010) 684–689
[50] Veltkamp, R.C., Hagedoorn, M.: State of the art in shape matching. In Lew, M.S., ed.: Principles of Visual Information Retrieval. Springer-Verlag, London, UK, UK (2001) 87–119
[51] Veltkamp, R.C., Tanase, M.: Content-based image retrieval systems: A survey. Department of Computing Science, Utrecht University (2002) 1–62
[52] Zalasinski, M., Cpalka, K.: New approach for the on-line signature verification based on method of horizontal partitioning. In Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L., Zurada, J., eds.: Artificial Intelligence and Soft Computing. Volume 7895 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 342–350
[53] Zitnick, C., Dollar, P.: Edge boxes: Locating object proposals from edges. In Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., eds.: Computer Vision ECCV 2014. Volume 8693 of Lecture Notes in Computer Science. Springer International Publishing (2014) 391–405