ON FUZZY REGRESSION ADAPTING PARTIAL LEAST SQUARES

Catalin Alexandru TANASIE[1]
PhD Candidate,
Doctoral School of Economical Informatics
The Bucharest University of Economic Studies, Romania

E-mail: catalin.tanasie@hotmail.com

Abstract
The usage stage of distributed IT&C applications – DIAs raises specific risks relating to the increased processing and usage strain related to live interactions. The incident categories and impact, as well as associated actors, are shown in order to serve as quantifying factors in the building of models aiming at quantifying their impact on distributed application reliability. A model aiming at extension impact assessment is built and details on the evaluation of the MERICS testing application are detailed. The component obsolescence is evaluated through an additional model and its impact on MERICS is shown alongside difficulties in identifying composing factors.
Key words: *distributed applications, interdependencies, MERICS, users, processes*

1. DIAs internal and external interdependencies

The interactions that describe DIA usage are controlled by authentication and authorization mechanisms that establish user identity and assign him with component or method-based access rights that enable the separation and differentiation of information control, with benefits to overall application security and data integrity. The following user role types are identified:

- *functional user roles*, associated to persons or processes that act in performing storage and computations on information as specified in the application's operational scenarios and defining the extent of operational method and data access, as well as managing content through differentiated query, insert, update and delete function access at database and table level, as well as associated read, write and delete file access rights; MERICS differentiates between the loading of images and video content and its review or testing with subsequent method access-driven, authorization-differentiated graphical interfaces through-out the presentation layer;
- *analytic user roles*, designed to manage authorized access to meta-information related to the functional domain in DIA usage - data mining, reports, building analytical structures such as OLAP cubes and reviewing results; MERICS defines analytical roles for both the definition and structuring of reporting based on input gathered by risk assessment functions and tools; they do not include access to

predefined reports that target unrefined operational data, leaving this prerogative to the functional roles;

• *technical user roles*, tasked with intermediating access to context and maintenance-related functions and tools as part of the distributed application components and deployment environment; administrators and operators interact with supporting DIA technologies in order to improve on performance, maintain the functioning status of the system and intermediate security tasks – including the creation and updating of user roles; MERICS is managed internally by the author and externally by the hosting service provider.

Defining security roles is done considering the user activity domain as related to DIA design and implementation specifications, which in turn determines the following categories and associated risks in *table 1*:

Table 1. Role-determined security risks

Name	Area	Description	Effects	Countermeasures
Excessive access granting	F	overextending user role access, appending existing credential rights instead of creating specialized new ones	data loss or unauthorized and unmanaged changes	new roles for new operational and data access combinations
Insufficient access granting	F	access restriction relies only on security criteria rather than including operational ones	DIA usage flexibility decrease, impossibility of finishing tasks	using user groups and encryption-based authentication mechanisms for sensitive areas
Improper use case mapping	F	failure in understanding security and operational requirements	communication and data quality loss	security role analysis and periodic reviews
Operational access	A	availability of altering mechanisms for the operational information that constitutes the basis for analysis	analytical output relevance loss, loss of operational information privacy	building automated information gathering and processing mechanisms
Undifferentiated analytical review access	A	insufficient delimiting in analytical information security implementation	privacy loss, productivity decrease through the building and usage of irrelevant reports in specific usage areas	classification of report security and information content loss effects
Technical personnel access	T	access to confidential operational and analytical information by virtue of technical skills and tools	information loss or altering	backup, auditing, delegation and separation of technical responsibilities
Unverified maintenance tasks	T	badly scheduled or un-reviewed maintenance jobs and actions on deployment tools and DIA components	interference with operational and analytical tasks, delays, data loss, unauthorized access	scheduling of maintenance, operational and analytical processes, documenting procedures

The *F,A* and *T* areas identify the functional, analytical or technical roles that constitute the risk source.

User responsibilities derive from roles by the addition of direct correspondence to operational use cases and the structure of the group or organization using the distributed application. They correspond to a mapping of the hierarchical structure of the organization and application usage roles, as defined in *figure 1* and the following generic model describing their composition.

Let set H define the hierarchical structure of the n users and persons that interact with the input or output of DIA components as specified by the functional use cases and described by

$$H = \{H_1, H_2, H_3, \ldots, H_k, \ldots, H_n\}, k < n$$

and set R of m items that define the roles associated to the usage of the distributed application components in the performing of tasks:

$$R = \{R_1, R_2, R_3, \ldots, R_i, \ldots, R_m\}, i < m.$$

The access to information and operations derived from the function and specifics of the position an employee or contributor has and its relation to neighboring nodes in the tree-like structure formed by describing these associations. Responsibilities define direct and indirect access and implicit influence of an actor on the content and form of the information operated upon by the system.

Let A_{H_i} be the set of roles mapped to node H_i, as determined by the specifics of operations performed. The responsibility $RESP$ of the associated user i hierarchical position does not limit itself to these, but includes the ones belonging to underlying hierarchical positions, defined by set $A_{rel}{}^i$ defining relating roles:

$$RESP_{H_i} = A_{H_i} \cup A_{rel}{}^i,$$

$$A_{rel}{}^i = \{RESP_{H_j} | H_j \text{ subordinate of } H_i, \forall i, j = \overline{1, n}\},$$

$$A_{H_i} = \{R_{i1}, R_{i2}, \ldots, R_{it_i}\}, \forall t_i = \overline{0, n-1},$$

$$R_{ij} \in R, \forall i = \overline{1, n}, j = \overline{1, t_i},$$

where:

t_i – number of associated roles for node H_i;

$R_{i1} \ldots R_{it_i}$ – roles associated directly to node H_i and corresponding to items in set R.

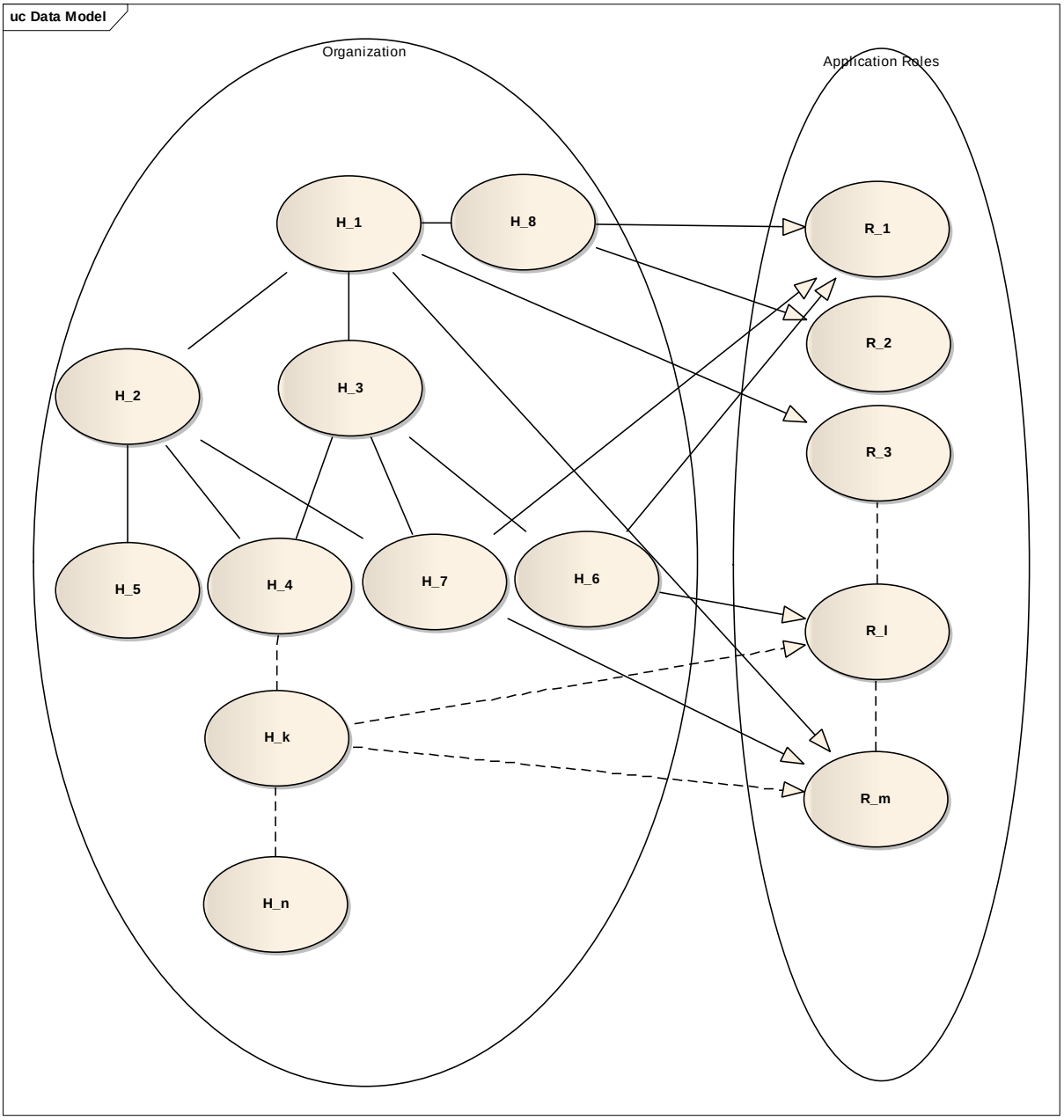


Figure 1. Organization hierarchy – application roles correspondence

Based on the previous model and information in *figure 1*, responsibilities for position node H_3, corresponding to user 3 has the following values assigned:

$$RESP_{H_3} = A_{H_3} \cup A_{rel}{}^{3}, A_{H_3} = \emptyset,$$

detailed as

$$\begin{aligned} RESP_{H_3} &= \emptyset \cup A_{rel}{}^{3} = \emptyset \cup \{RESP_{H_4}, RESP_{H_5}, RESP_{H_7}\} \\ &= \emptyset \cup A_{H_4} \cup A_{rel}{}^{4} \cup A_{H_5} \cup A_{rel}{}^{6} \cup A_{H_7} \cup A_{rel}{}^{7} \\ &= \emptyset \cup A_{H_4} \cup A_{H_m} \cup \emptyset \cup A_{H_5} \cup \emptyset \cup A_{H_7} \cup \emptyset \\ &= \emptyset \cup \emptyset \cup \{R_{m1}, R_{m2}\} \cup \{R_{71}, R_{72}\} \cup \{R_{61}, R_{62}\} \\ &= \emptyset \cup \emptyset \cup \{R_k, R_m\} \cup \{R_1, R_m\} \cup \{R_1, R_k\} = \{R_1, R_k, R_m\} \end{aligned}$$

As observed, even if user 3 has no direct role associated, he inherits from the relationship defined by links in the DIA hierarchy. The assignment of responsibilities

introduces the problematic of defining and delimiting hierarchical positions and associated workloads, in order to correctly map roles to operational and analytical functions.

Distributed application components are separated by platform, role, physical location and security concerns in sections which interact within platforms and communication media shared by multiple systems. The reliance on common information structures and messaging channels impacts on the performance and availability of methods and creates the need for assessing the impact of incidents and improperly functioning platform components in the performance of the application. The *interdependency* property, defined as the degree in which the operational status and output of a component influences the activity of another.

Internal interdependencies characterize DIA modules and tools within the same application, relating to communication, synchronous and asynchronous operation, security and incident effects, as well as the impact of damaged or invalid information in the functioning and output relevance for later usage. MERICS introduces dependencies of different magnitude as the architectural layer increases, with persistence isolated and secured from synchronicity or damage propagation as compared with the service or presentation layers.

External interdependencies define interactions with outside modules, across communication channels whose traffic is not under the supervision of the DIA operating parties and are accessible to a various degree to the general public. In addition, it includes the aspect of deployment platform failures or hardware performance as a factor that affects the functioning of components. MERICS, the distributed application used as a testing platform in risk factors identification, as well as risk assessment model evaluation, relies on the deployment platform specifics – hardware, software instruments – in the overall performance, with impact on the timing of synchronous methods and susceptibility to security threats.

The items shown in *tables 2, 3* and *4* are selected to reflect on their importance in the usage of distributed applications, alongside a description of effects, counteractions and the practical implementation of these, numbered as follows: *MERICS.TEST.Desktop* (1), *MERICS.WEBAPP* (2), *MERICS.LOGICAL* (3), *MERICS.OPERATIONAL* (4), *MERICS.COMMON* (5), *MERICS.DataOperations* (6), *MERICS.AUTHENTICATION* (7), *MERICS.WCF* (8), *MERICS.Service* (9), *MERICS.ANALYTICAL* (10) and *MERICS.CONTEXT* (11), as well as the operational (12) and analytical (13) databases.

The communication channels represent a vulnerable component of distributed applications through their susceptibility to attacks and the unavailability of details relating to their operating status and performance. The information transferred across them undergoes two separate and complementary processes, as formats in emitter and receiver entities are aligned and security-enhancing procedures are performed. *Table 2* details on the risks that induce lower operating quality and increase the time needed to process requests.

Table 2. Communication risks

Risk	Description	Effects	Counteractions	MERICS
Unsynchronized data contracts usage	outdated WSDLs, failure to communicate changes, changes in the optional status of method arguments	errors, incomplete parsing of information, deprecated methods and attributes usage	documenting and communicating changes on the emitter side, periodic validation of message format on the receiver side; usually appears in public, general-use Web services – weather, exchange rates	(5), (8), (9)
Improper type formatting	changes in encoding, length, content appearing in formatting and encoding or decoding	data loss errors, loss of information quality, delays due to recasting	validating hardware and software compatibility in communication actors	(8), (9)
Incompatible complex structure usage	using custom-built structures that rely on incompatible data types or which are not described by data contracts	errors and information loss due to decoding failure	including complex type definition and encoding in service description files	(5), (8), (9)
Variances in endpoint security	changes in security levels through-out the communication components	security validation errors, authentication failures	assessing the impact of unilaterally increasing or decreasing security	(7), (8), (9)
Message delays	time-outs in receiver communication endpoints	errors and delays in task processing, loading of memory for queued messages	asynchronous methods, alternate, interchangeable role modules	(3), (4), (8), (9)

The nature of DIA component interactions is a factor in the measuring of incident susceptibility, as the context of their operations raises risks with respect to data quality, module availability and processing load. The interaction between components and dependency on timed actions constitutes a criterion in the definition of asynchronous and synchronous operations.

Asynchronous processes, shown in *figure 2*, bottom section, operate on information without having to relate on external output in the finishing of tasks. Data formatting, as well as scheduled jobs in operational and analytical databases belong to this category. In the course of their execution, they do not require or rely on data changes triggered by other components. They are less susceptible to errors relating to informational quality or validation than their counterparts. MERICS implements asynchronous operations primarily at database level, as the formatting and export of operational database records for data mining purposes is a primarily technical task done without regard to the quantity or content of information, within predetermined specifications and using known filters. *All input information is known at*

the moment of execution. If processes communicate, response information is used outside the context of the task.

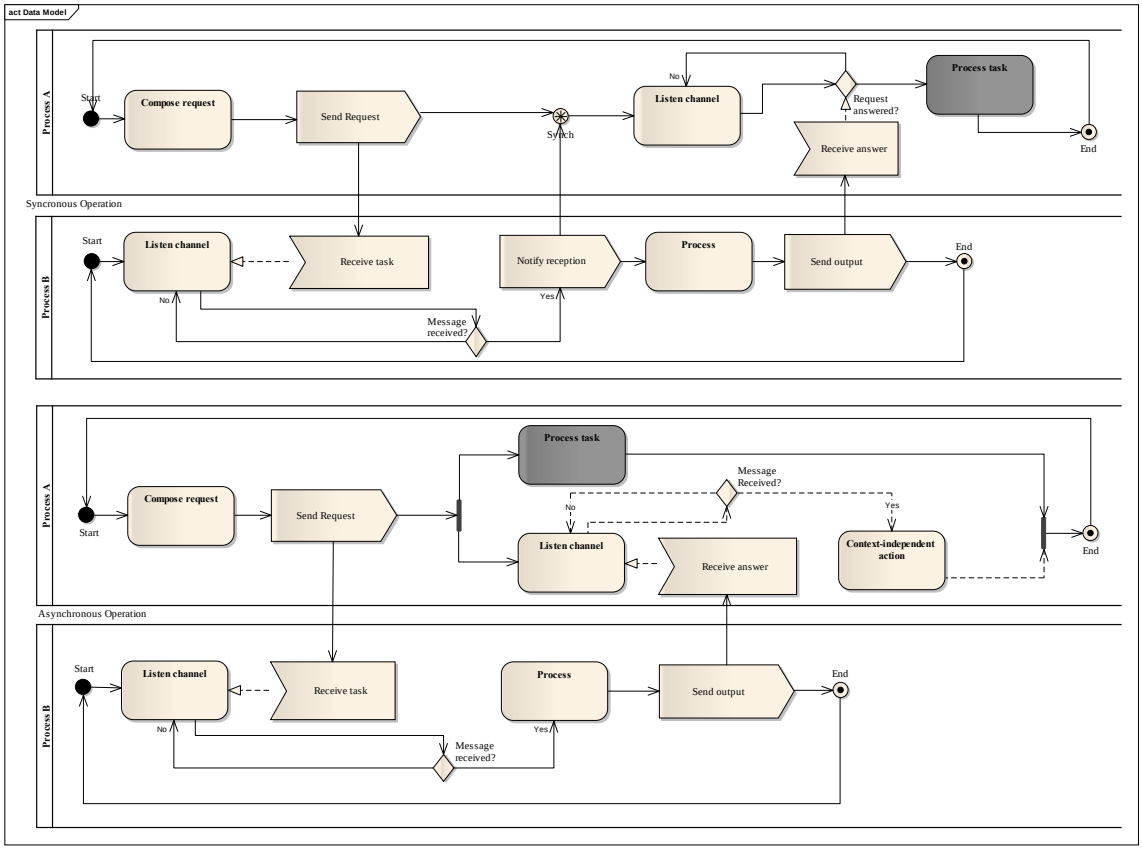


Figure 2. *Synchronous and asynchronous processes*

Synchronously operating processes, shown in *figure 2*, top section, depend on others in the solving of tasks, and whose output is affected by the order of interactions and are susceptible to incidents that derive from the timing and information dependencies in these steps. *Not all information is known a priori*, as opposed to the other category, with consequences on the order and timing of steps. *Table 3* identifies the incidents that synchronous operations involve, as well as the MERICS components implementing countermeasures.

Table 3. Synchronous process incidents

Incident	Description	Effects	Counteractions	MERICS
Time-outs and component unavailability	failure in receiving the answer within a given period, either arbitrarily chosen or predetermined	outcome quality and availability	alternate services, extending time-out intervals	(3), (4), (8), (9)
Excessive request strain	overloading of a component's capabilities by request number	delays in processing requests, errors in emitter and sender components	extending hardware capabilities, adding similar components, load balancing software controllers	(3), (4), (10), (11)
Cascade effect propagation	delay time builds up as multiple components are affected	in services that share communication channels and message queues, unrelated incidents cause failures in properly functioning exchanges	using multiple communication channels for critical tasks, prioritizing and excluding underperforming items from the queue	(3), (8), (9), (11)
Brute force	limitations in timing increase attacks damage	unavailability of communication	Implementing communication pattern detectors and additional service components	(7), (8), (9)
Impersonation	limits in response time and available security protocols increase the likelihood of successful security breaches	data theft and altering due to failures in detecting attacks over small periods of time	Using pattern detectors, switching to asynchronous messaging in data whose exchange is not time-critical	(7), (11), (10)

Autonomy is the DIA component level correspondent of asynchronous processes, defining modules that act independently of the status and informational content of other runtime components, relying on input information that is already present within the system. However, this feature does not exclude vulnerabilities deriving from the quality of both input and output data, as other processes influence the relevancy of results. Additionally, autonomy is not required to be mutual, as DIA usage includes scenarios in which these components serve as real-time input providers for others, often within synchronous jobs. *Table 4* details on vulnerabilities, effects and MERICS components implementing software features minimizing impact on the application.

Table 4. Autonomous component vulnerabilities

Vulnerability	Description	Effects	Counteractions	MERICS
Data relevancy	unavailability of real-time quality checks	storage and processing of improperly validated information	implementing validation controls in both input and output information	(1),(2),(4), (6),(12), (13)
Incident communication	failures are not detected instantly by other components in the system	reliance on improperly functioning components	implementing auditing and fault detectors	(7),(8),(9), (11),(12), (13)
Specialization	autonomous components act in predetermined, inflexible process areas with a high degree of specialization	limited reliance on autonomy as protection against errors and security threats	extending impact by implementing asynchronous computation tasks in autonomous components	(3),(4)
Error detection	data loss and altering is not immediately detected outside autonomously operating components	improperly formatted, invalid information in interdependent components that process information at a later time	synchronous fault detectors, validation in all interacting components	(1),(2), (3),(4), (9),(11)
Improper maintenance	improper functioning is not readily understood or detected by maintenance crew	derived from the high specialization, it affects component and process availability	documentation, training, separation of tasks within technical usage areas	(3)

Information flow in DIA-mediated tasks is dependent on the synchronization of components and availability of input for each successive step in computation. The specialization of DIA modules, beneficial to the speed and quality of output, increases incident risks due to the dependencies it imposes, as the system components do not posses all available information and algorithms to provide answers, relying on collaboration to achieve the completion of jobs. Considering this property, deficiencies in information synchronization include:

- *reduced contextual awareness* in multi-system collaboration; the users of an application are performing specific tasks, and may not be completely aware of the relevancy and global positioning of the specific stage they mediate, leading to decreased information quality as the input is not contextually validated and security vulnerabilities by the failure to protect data as its sensitivity is unknown;
- *information quality deficiencies*, as collaborating components rely on previous stages in validating information and take its correctness for granted; an algorithm for assessing component performance is limited to factors inside the analyzed methods; in MERICS this feature created problems in both operational and analytical modules, as the need quality of output is dependent on the entirety of actions performed as part of an use case; adding validation controls

and cross-system analytical factors reduces the incidence of incorrect information processing.

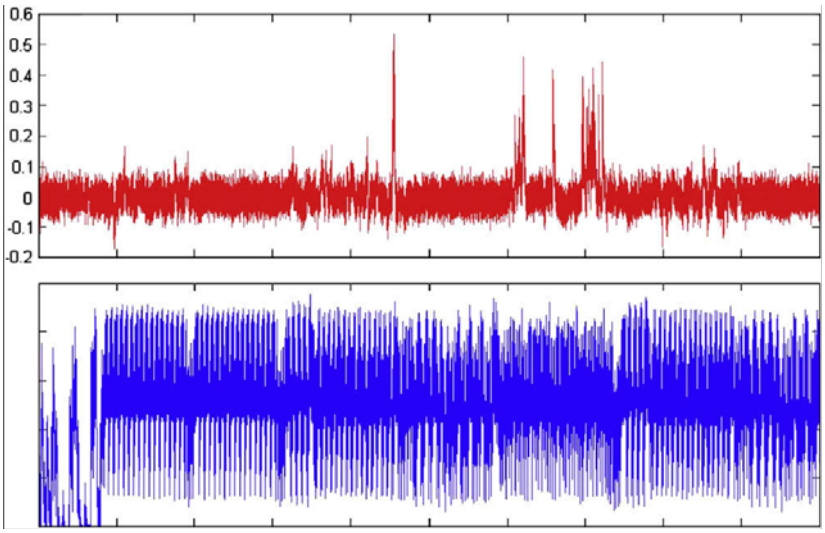
2. DIA maintenance risks

Maintenance relates to the manual tasks and processes that serve the optimum functioning of DIA components and communication channels. It is performed by technical users, administrators and external parties, as well as by means of automated repository and memory cleansing, message flow refining, load balancing and caching operations. It contains two separate areas of interest as relating to the target of the jobs performed – hardware and software.

Hardware maintenance groups together actions that aim at the updating and ensuring a proper running state for devices on the DIA deployment platform. Considering risks developed as part of the maintenance processes, hardware management targets the avoidance or minimization of:

- *power failures*, with energy backup systems and recovery monitoring; the purpose of installing alternative generators ranges from ensuring an interval for saving session and operational information before performing a controlled shutdown, in the lower extreme, to the indefinite ensuring of the power supply in a transparent transition that does not affect user activity;
- *hardware component failures*, with repair or replacement options in situations where recovery is impossible; servicing, intermediate backup systems, multiple interacting units similar to parallel processing ensure the minimization of incident effects; documenting on recovery procedures and communicating vulnerabilities, as well as tracing the source of the incidents helps reduce the inherent component downtime or increased strain on similar ones in DIA usage;
- *data loss* – potentially damaging to the relevancy and availability of information, it relates to failures in storage instruments – hard-disks, backup tapes, mobile devices; prevention through backup and the subsequent maintenance of versions and copies by specialized companies or through internal resources, recovery procedures for damaged disks, fire prevention for storage rooms, ensures the lowering of costs induced by missing or irrecoverable information; the budget for such procedures varies depending on the activities that distributed applications manage;
- *security* – brute force attacks, unauthorized access, data handling leaves traces in the hardware components runtime indicators – power usage, temperature, sound; AES encryption information is gathered by means of viewing patterns in electrical voltage as blocks of data are encrypted every nine steps and every cipher item leaves a distinct signature – figure 2 lower section, with the attacker able to identify specific patterns and gradually identifying components through successive trials; in a similar fashion, maintenance operators and processes trace the hardware signature of attacks as part of routine component status surveillance – figure 2, upper section (1).

Figure 3. AES encryption algorithm attack hardware signature (1)



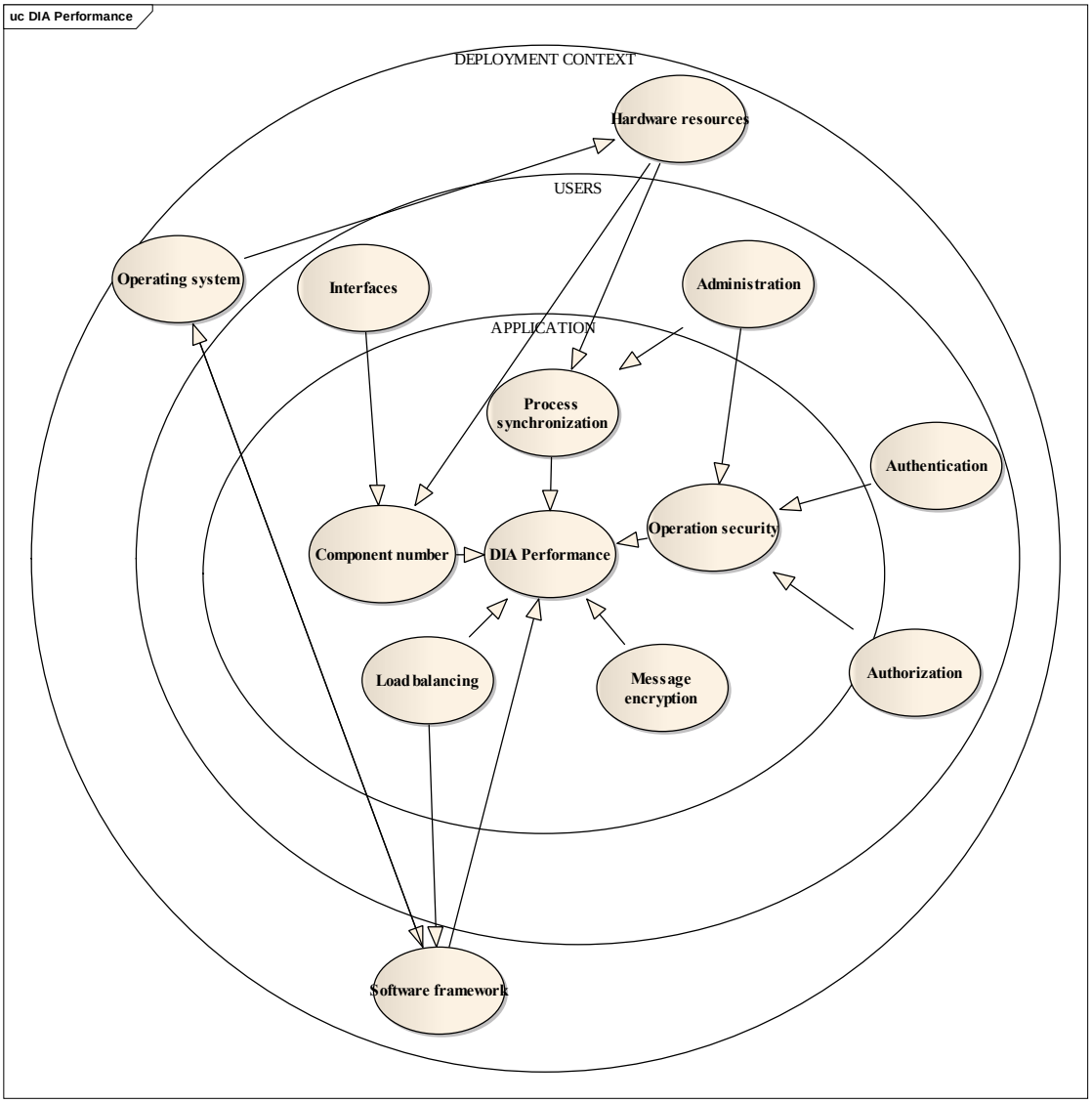


Figure 4. DIA performance dependencies

Information is exchanged between DIA components by means of messaging conduits, formatting and encrypting data at one end, as well as validating credentials and recomposing the programmatic entities at the other. These operations are influenced by user access and the operational status of the channels and endpoints. In concurrent, multiple source DIA interactions, credentials and message content serve as prioritizing factors, as well as the first layer of defense against security breaches. Maintenance tasks relate to the management of:

- *user credentials*, as over the lifespan of the application the identities of the accessing persons and processes change, as well as the validity of input – passwords and digital certificates expire, users move inside and outside the DIA-operating organization, component security requirements change with the diminishing or increasing of risk factors for specific tasks; maintenance technicians and scheduled processes ensure the periodic updating of database and operating system access, especially in user accounts with administrative or confidential clearance;
- *message queuing review and configuration*, with surveillance of channel load, incidents and configuration of bandwidth for the insurance of efficient and time-efficient communication, especially with respect to synchronous processes; prioritizing messages and incidents based on severity scales configurable through administrative interfaces; the *MERICS.CONTROL* module delegates tasks acting in part based on preconfigured component and message priority, with the possibility of runtime reevaluation.

The direction of the dependency indicators in figure 4 as opposed to the context also indicates the coverage area of the effect, with inward referencing arrows indicating specificity to the described process, and outward the generalness of the property. Considering a graph representation of the interactions, the roads between nodes indicate the chain of dependencies; the software framework, as described in the approach, impacts DIA performance in two ways – directly, through processing support, protocols and standards, and indirectly, by influencing and being influenced by the operating system, which in turn defines the hardware resources usage and indirectly the maximum component number, with direct effect on DIA performance.

The performance of distributed application components is affected by risks deriving from vulnerabilities and interfering in successive layers as related to the usage environment, shown in figure 3:

- the deployment context, associating hardware and software support components as well as the technologies that form the basis of the application runtime execution; operating systems and software framework choices affect each other and the system's performance; MERICS uses cross-platform Microsoft technologies, with the optimization of their interaction relevant in isolating external interference in determining operational and assessment model behavior;
- the usage context, with authentication and authorization, as well as operational administration and interface design impacting on the amount of resources used by DIA modules; the security level, encryption protocols and number of external interactions affect the performance and computing efficiency; MERICS separates

the endpoints based on the risk assessment values, with effects in the optimization of resource sharing;

- the application, with component number, process control, security and encryption algorithms implementation influencing the amount of computation power needed for usage; MERICS implements operational control, multi-threading, task separation and ordering, as and analytical evaluation of performance indicators, with continuous optimization for underperforming algorithms.

CONCLUSIONS

Incident prevention ensures the minimization of frequency of occurrence, as well as the cost of recovery and revision of application components. In situations where prevention fails or is not envisioned and incidents occur, assessment and recovery protocols ensure the lowering of damage done through direct and indirect costs. Table 5 identifies the steps as envisioned during the development and usage of the MERICS distributed application.

Table 5. Disaster recovery steps

No.	Step	Issues	Actors
1	Identification	determining the source, security break, affected components	operational users, developers, maintenance crew, system administrators
2	Stopping of malicious activity	action effectiveness, difficulties in eliminating all attack routes	administrators, operational users
3	Removal of damage	restarting affected components, recovering lost or tampered information	functional and database administrators, operators
4	Behavior description	area of incidence, technical or logical vulnerability	users, business analysts, system designers, developers, testers
5	Assessment of effects	choosing assessment models, risk budgeting, cost valuation	users, operational management,
6	Application updating	extending functionalities in affected components, improving security algorithms and procedures	system and security designers, developers
7	Testing	validating changes, reproducing incident scenarios	developers, testers, users
8	Deployment	replacing faulty components in the live usage environment	Functional administrators, testers, users
9	Documentation	Evaluating impact and documenting effects, patterns of occurrence and response	users, business analysts, designers, developers, testers, management

The completion of first three steps of the procedure defines the cost impact of the incident, as the timing and tools available in detecting and counteracting threats and failures in the application's components influence their span and effects.

The damage removal stage in disaster recovery procedures includes the resubmitting and reprocessing of pending requests and tasks at the moment of incident occurrence. Factors in the prioritizing of jobs derive from the following aspects:

- the severity of the request in what concerns the importance of output delivery speed in the quality of the response; real-time information such as exchange rates and stock exchange quotations lose their relevance over small periods and must be processed by alternate modules; MERICS prioritizes information exchange through the usage of its control role component, as well as reevaluation of the delegation mechanisms through input from MERICS.ANALYTICAL and associated database;
- the identification of erroneous messages following the same pattern that caused the error, if the structure or content of the communication was the source of the vulnerability, as well as the identification of security threats related to security incidents, in case the attacker forces his entry into the system by more than one communication item.

The reprocessing capacity of the system is improved by the implementation of role-interchanging components, which are available for task delegation in case the functionality of one module is disturbed.

References[2]

1. Adams, C., Farrell, S.; **Internet X.509 Public Key Infrastructure: Certificate Management Protocols**, Network Working Group, March 1999, http://tools.ietf.org/html/rfc2510
2. Konstantinos Markantonakis, Michael Tunstall, Gerhard Hancke, Ioannis Askoxylakis, Keith Mayes – **Attacking smart card systems: Theory and practice**, Information Security Technical Report, Nr. 14, 2009, pg. 46 – 56, ISSN: 1363-4127
3. Nagpal, R.; **Simple guide to digital signatures**, Asian school of cyber laws, 2008, www.asclonline.com
4. Nissanke, N.**; An integrated security model for component-based systems**, Emerging Technologies and Factory Automation, 2007 ETFA IEEE Conference on Emerging Technologies and Factory Automation, pp. 638 – 645, ISBN 978-1-4244-0825-2
5. Tilborg, Henk C.A. van; Jajodia, Sushil, **Encyclopedia of Cryptography and Security, 2nd edition**, 2011, Springer Reference, 1435 pp., ISBN 978-1-4419-5905-8

[1] *Catalin Alexandru TANASIE is a graduate of the Faculty of Cybernetics, Statistics and Economic Informatics within the Bucharest University of Economic Studies, the Economic Informatics specialization, 2007 promotion. Starting the same year he attended the Informatics Security Master in the same institution, and is currently a PhD student at the Doctoral School within the Bucharest University of Economic Studies. He has concerns in the field of distributed applications programming, evolutionary algorithms development, part of the field of artificial intelligence - neural and genetic programming. Currently he works as an application designer in a financial institution, in areas concering the development of commercial applications.*

[2] Codification of references in text:

[1]	Konstantinos Markantonakis, Michael Tunstall, Gerhard Hancke, Ioannis Askoxylakis, Keith Mayes – **Attacking smart card systems: Theory and practice**, Information Security Technical Report, Nr. 14, 2009, pg. 46 – 56, ISSN: 1363-4127
[2]	Adams, C., Farrell, S.; **Internet X.509 Public Key Infrastructure: Certificate Management Protocols**, Network Working Group, March 1999, http://tools.ietf.org/html/rfc2510
[3]	Nagpal, R.; **Simple guide to digital signatures**, Asian school of cyber laws, 2008, www.asclonline.com
[4]	Nissanke, N.**; An integrated security model for component-based systems**, Emerging Technologies and Factory Automation, 2007 ETFA IEEE Conference on Emerging Technologies and Factory Automation, pp. 638 – 645, ISBN 978-1-4244-0825-2
[5]	Tilborg, Henk C.A. van; Jajodia, Sushil, **Encyclopedia of Cryptography and Security, 2nd edition**, 2011, Springer Reference, 1435 pp., ISBN 978-1-4419-5905-8