JOURNAL OF APPLIED QUANTITATIVE METHODS

Quantitative Methods Inquires

EVALUATING SECURITY THREATS IN DISTRIBUTED APPLICATIONS LIFESTAGES

Catalin Alexandru TANASIE [1]

PhD Candidate,
Doctoral School of Economical Informatics
The Bucharest University of Economic Studies, Romania

Application Designer,
Raiffeisen Bank S.A., Bucharest

E-mail: catalin_tanasie@yahoo.com

Eduard Emanuel HERTELIU [2]

PhD Candidate,
Doctoral School of Economical Informatics
The Bucharest University of Economic Studies, Romania

E-mail: emanuel.herteliu@gmail.com

Abstract:

The article starts with the classification of security threats as related to the context of operating distributed IT&C applications – DIAs, as concerning users, processes and the information exchanged. Security risks induced as part of the analysis, design and development phases of distributed application building are detailed alongside proposed countermeasures. A model addressing security element interaction is presented and details on its implementation as part of the authentication module of the model testing and refining application, MERICS, is shown.

Key words: *distributed applications, information, security, risk, models, metrics.*

1. Classifying DIA security risks

DIA security relates to the interaction of actors, data messages, operations and contextual parameters in ensuring the privacy and operability of the system's informational content, operations and contextual parameters. The following constitute elements of risk in the context of the system's security:

- *information*, the content operated on by computing instruments in the processes that characterize operational DIA modules, alongside the output obtained from the underlying methods; risk sources include the loss of privacy for transferred or stored data, unauthorized acquisition of application and context of operation parameters and authentication credentials;

- *users*, which determine the quality and content of messaging and operations in DIA activities, impacting risks by the uncontrollable nature of their actions as viewed in the application's context – no security protocol is able to prevent the unauthorized disclosure of credentials; measures to prevent or minimize damage, include the updating of security tokens to levels which are difficult or impossible to replicate or know without context-dependent input – dynamic passwords, certificates that rely on third parties or cryptographic instruments; in situations where complex authentication is impractical – public information portals, virtual libraries, news networks – the prevention of incidents relies on auditing and preparing a set of runtime automatic assessment and threat prevention procedures – logging out suspect users, shutting down or refusing the initiating of new sessions past a predetermined threshold;
- *administrative processes* – relating to the set of maintenance tasks and actions done inside or outside the operating context of the application – assignment of user authentication credentials, monitoring of processes and communication channels, ensuring the operational status for the application's hardware and software platforms, logging and treating errors, managing and prioritizing tasks, interacting with databases and file systems in ensuring the proper functioning of querying and persistence-related functions; security risks relate to the implicit potential of damage resulting from the preferential operational and informational access that accompanies administrative roles and tasks;
- *communication* – the generating, transfer and reading of information through messaging tasks, over public and private networks; the encoding, encryption and decryption capabilities of the communicating parties determine the security potential of the packages of transferred data; risk sources include the number and operating context of communication channels, data encryption capabilities, asynchronous operation features, messaging or transfer paradigms – immediate or synchronous, queue-enabled as in Service-Oriented Architecture; the relevancy of the latter is due to the *time*-dependent nature of cryptographic tools – few, if any, of these are immune to dictionary-based attacks or exhaustive key searches over extended periods of time in the context of an exponential increase in processing power over the past decades, their security deriving from being able to prevent information leaks within relevant operational time frames – an attacker who deciphers dynamic, runtime-generated encryption, financial content months after the message was sent is not able to use this information in impersonating communication parties.

In considering the measures that the DIA interactions actors, involved in the design, development and usage of distributed applications, need to take in order to generate an accurate model for security threat pattern detection and identify targeted and improperly constructed components, the structuring of vulnerabilities and associated risks is required. The following constitute security risk classification criteria in assessing DIA vulnerability levels:

- *context of appearance*, with risks originating inside or outside the construction and usage domain of the system; when assessing a Web service, the incidents originating in database or file system information disclosure constitute *internal* causes for the associated costs, directly traceable to the improper development of cryptographic instruments; security errors due to loss or mismanagement of user credentials are

determined by both the improper design of crisis procedures and techniques and *external* factors including deployment and security platforms owner and maintenance crew;

- the *effect* on the application domain, with *operational* risks relating to incidents that target components, communication channels or repositories by preventing their proper functioning, either through the direct interaction and alteration of parameters, or indirectly by exploiting vulnerabilities in their design – brute force attacks, preventing synchronous messaging in interdependent components; the second subcategory is formed by *informational* risks, relating to the altering or unauthorized accessing of data structures, allowing the attacker gains by exploiting the discovered parameters and confidential content;
- *frequency and damage*, which together form the measure of costs to the operator and users of the system; in the generic economic context of a limited number of resources and infinite number of needs, the latter are organized and prioritized so that a maximum level of satisfaction is obtained; translating to risks, the losses due to the occurrence of unwanted operations are minimized by implementing supplementary controls and procedures.

Actors involved in the design and development of the distributed application, shown as they interact in *Figure 1*, contribute as factors to the constriction and evaluation of risk assessment models. *Table 1* shows the origin and relevant DIA lifecycle phase for each influencing operation, along with associated risk and nature of the model's input and/or results by description of countermeasures.

Table 1. Analysis, design and development actor-induced security risks

Actor	Phase	Risk	Countermeasures
User	Pre-analysis	improper evaluation of security threat potential	assessing the potential losses by identifying and valuing operational and informational damage
User	Analysis	incomplete specification or knowledge of required activities and information sensitivity	security-oriented valuation of DIA content and operations by analysts
Analyst	Analysis	inaccurate understanding of security tasks as specified by the operational user, especially concerning large, interlinked activity sets	formalizing and documenting the requirements and acquiring cross-market information on relevant threats
Analyst	Design	improper specification of security constraints	inclusion of development and design parties in the evaluation of requirements and techniques
Designer	Design	technological choice limitations due to security costs	assessing risk frequency and potential damage in distributing security controls and tools between DIA components
Designer	Design	over extensive, interdependent security technique specifications	evaluation of cross-component impact of security protocol choices
Developer	Development	improper implementation of security controls	assessing threat levels for each DIA component type and communication channel
Developer	Development	insufficient auditing tools for operations and data structures	analyzing incident target area and supplementing information change monitoring

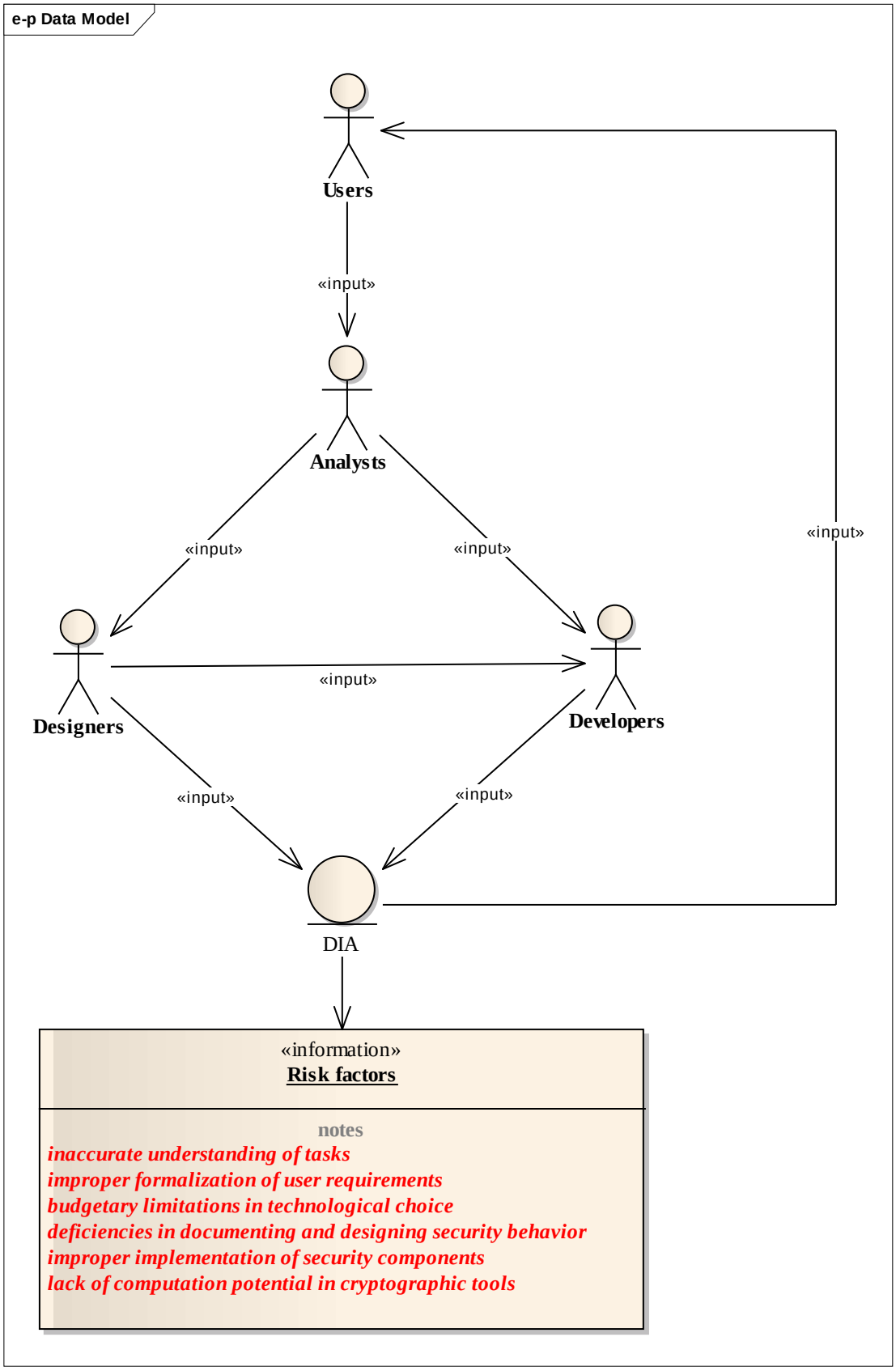

Figure 1. Analysis, design and development actors as risk factors

The influence of security incidents on the costs of maintaining and running DIAs is directly proportional to the exposure of the system's endpoints, dimensioning of sessions and user base, as well as the nature of operation. Sensitive processes and messaging require the addition of supplementary steps as part of the operational methods. The variance in distributed application span and coverage, as well as areas of usage, requires the establishment of universally valuable, cross-industry measuring units of cost. MERICS, the distributed software system used in factor analysis and risk assessment model evaluation, is designed with auditing controls that measure *timing* and *computing* resource strain. The first element is defined as the relation between the number of hours that users and developers, as well as automatic processes, require in order to prevent or remove effects as opposed to global indicators of value for the distributed application's owning organization.

2. Information security risk elements

Securing access to information as part of inter-component communication and persistence-related operations requires the formalization of interactions within DIA activities, as well as the development of procedural mechanisms that manage and audit authentication jobs.

Let $UP = \{U_1, \ldots, U_n, \ldots, U_m, P_1, \ldots, P_n, \ldots, P_m\}$ of n users and m processes, represented for simplicity purposes as a set of entities belonging to the two categories, $\{e_1, \ldots e_{n+m}\}$, requiring authentication by means ranging from the simple providing of a password to dynamic, context-dependent token information and cryptographic operation-enabled credentials such as digital certificates.

Let the function $pass_j(e)$ describe the status of validity for user of process e with regard to feature j in the authentication criteria, as follows:

$$pass(e) = \begin{cases} 0, & \text{authentication factor not present or compiled} \\ 1, & \text{authentication factor present or compiled} \end{cases}$$

The granting of access to the presentation and service layers of the distributed application is described by function $acc()$ described as a product of s conditions as follows:

$$acc(e_i) = \prod_{j=1}^{s} pass_j(e_i), \qquad i = \overline{1, n+m}, \qquad j = \overline{1, s}$$

where

- e_i – element i in set UP; authentication requester – user or process;
- j – number of conditions that the authenticating entity must fulfill in order to be granted access to the functions of the distributed application;
- s – Number of requirements applied to the authenticating entities.

Considering the previously presented format, the possible values of the $acc()$ function describe the same range as function $pass()$:

$$acc(e_i) = \begin{cases} 0, & \text{one or more authentication conditions are not fulfilled by } e_i \\ 1, & \text{authentication factor present or compiled by } e_i. \end{cases}$$

The logic of authentication operations is described as a repetitive block of the form shown in *figure 2*. The activity diagram defines steps in evaluating entities as part of the MERICS.AUTHENTICATION module, with regard to factors as presented in *table 2*.

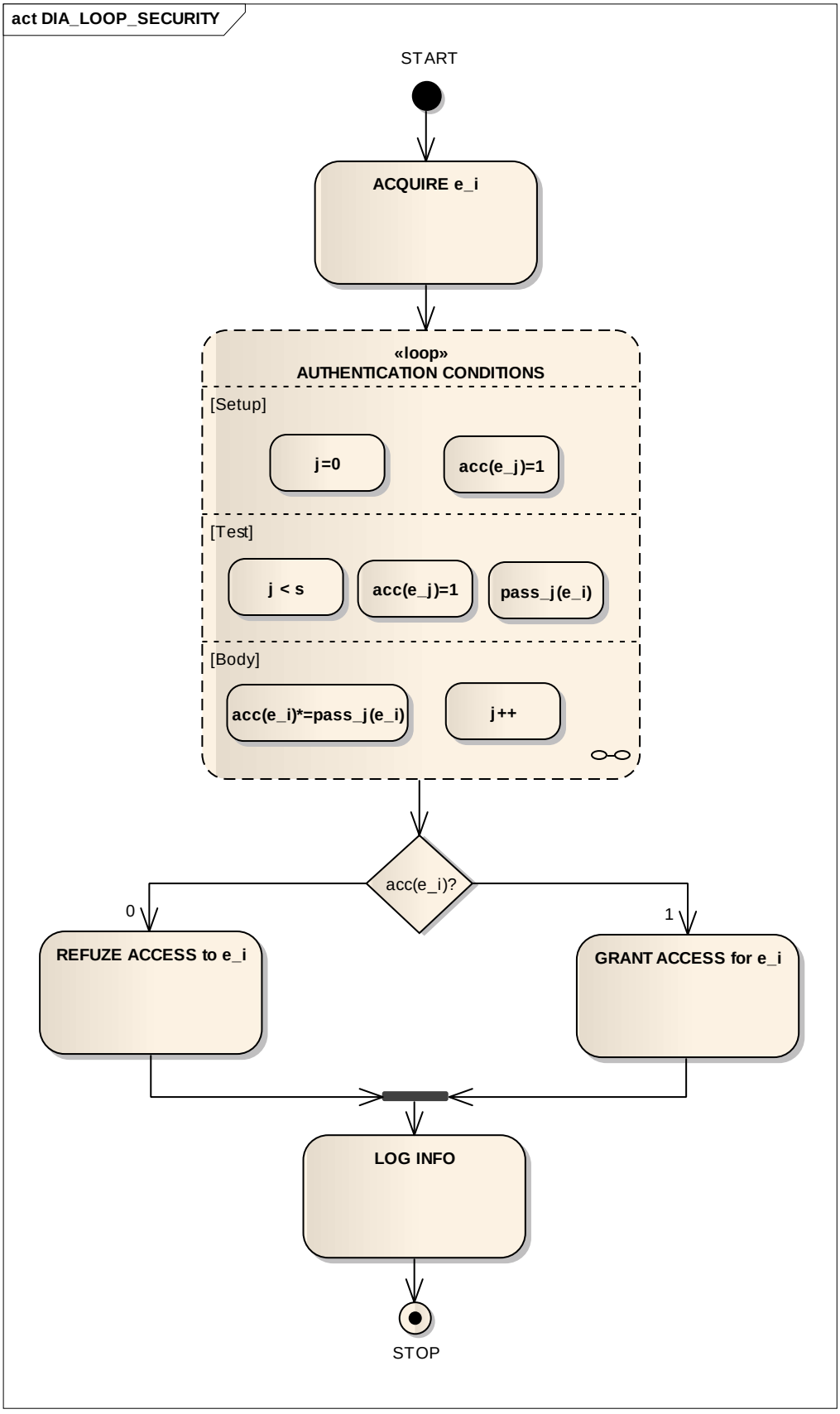

Figure 2. MERICS.AUTHENTICATION

Table 2. Authentication criteria as part of *MERICS.AUTHENTICATION*

Criterion	Description	Test order
Credentials	validation of the presented security credentials by comparison to an internally administered user credential list	1
Black list	compare the specified credentials to a predefined list defining reported sources of incidents, dynamically updated based on observed behavior and effects of DIA interactions by the users and processes	2
Groups	the user group that the currently evaluated user belongs to	3
Roles	the roles that define permitted operations and component access for the assessed entity	4
Operations	the requested action as compared to allowed interactions	5
Time	the time of the request, used to determine the relevancy of the request	6
Location	the network location of the requester	7
History	derived from the blacklist, time and location criteria, the history of access by the evaluated entity is used to determine the authenticity of the request	8

The following two cases constitute a hypothetical scenario for the *MERICS.WEBAPP* module communicating with MERICS.WCF and evaluated by *MERICS.AUTHENTICATION* – *table 3*.

Table 3. Evaluation as part of *MERICS.AUTHENTICATION*

Source	Target	Operation	$acc(e_i)$[1-8]	Resolution
MERICS .OPERATIONAL	***MERICS .DataOperations***	***query***	1*1*1*1*1*1*1*1	**1:ALLOW**
MERICS.WEBAPP	**MERICS.WCF**	***query***	1*1*1*1*1*1*1*1	**1:ALLOW**
MERICS.WEBAPP	**MERICS.WCF**	***delete***	1*1*1*1*1*1*1*1	**1:ALLOW**
MERICS .LOGICAL	**MERICS.WCF**	***query***	1*1*1*1*1*1*1*1	**1:ALLOW**
MERICS .LOGICAL	**MERICS.WCF**	***delete***	[1*1*1*1***0**]*1*1*0	**0:DENY**
MERICS.WEBAPP	***MERICS .DataOperations***	***query***	[1*1*1***0**]*0*1*1***0**	**0:DENY**

An additional source of vulnerabilities consists of the transfer of information between components as part of DIA activities. The specific separation of varying activities and role-based or geographical separation increase the incidence of cross-component communication as compared to other software application paradigms. *Figure 3* details on the buildup of risk augmenting factors, starting with the first stages of an application's lifecycle, as a graph detailing on dependencies as defined, in order of precedence, by:

- *requirements*, influencing the activity domain of DIA interactions, as well as the user roles that are defined for their management; identifying the vulnerabilities early on reduces costs; the lack of operational information in development environments tests and the restructuring of development tasks to account for the issues allow for

the validation and improvement of the application's components before incidents occur;

- *users*, whose number is a defining characteristic in the definition of operational computation and storage requirements, as well as frameworks chosen and extent of

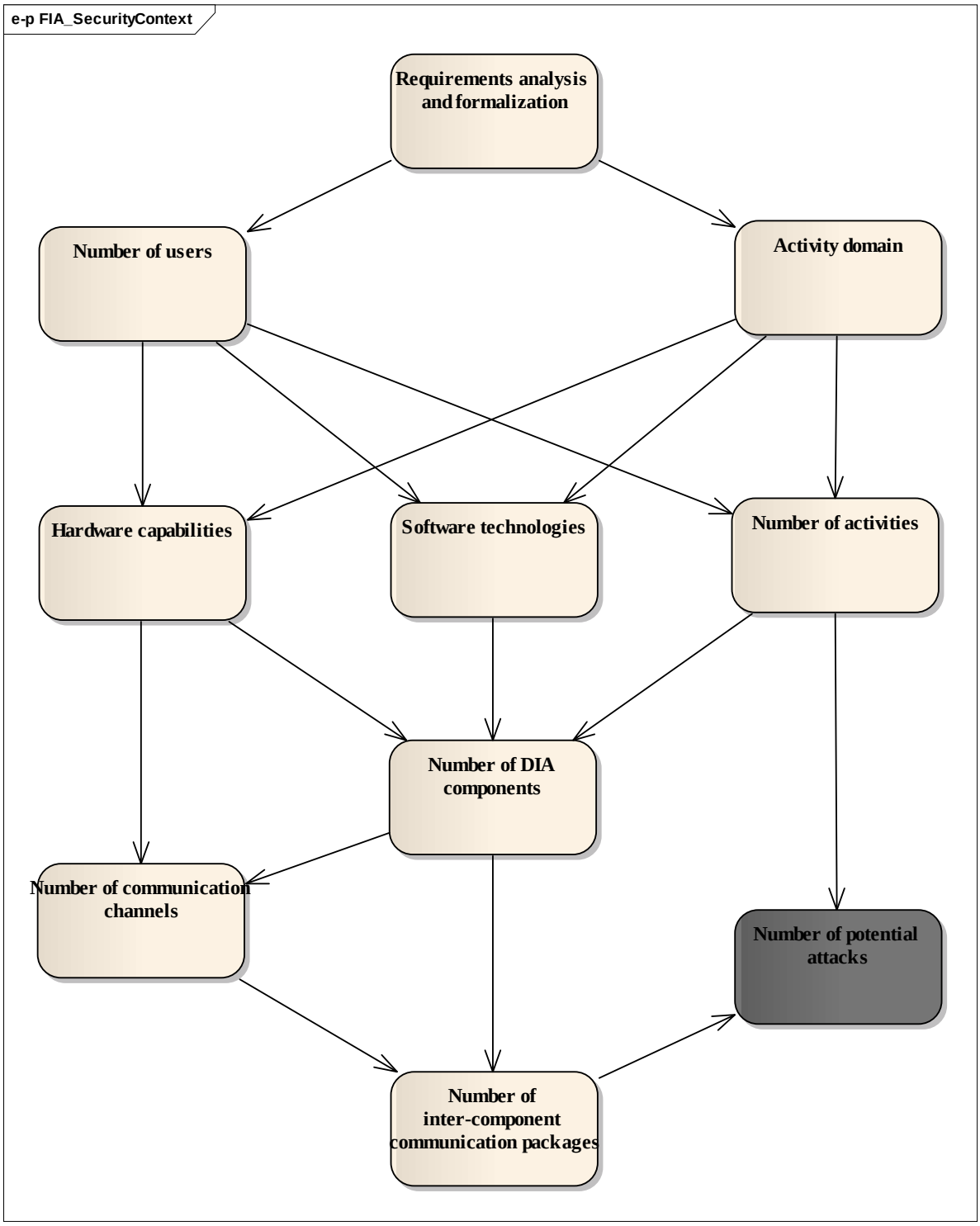

Figure 3. Information transfer threat potential factors

activities intermediated by the application; the nature of user activities identifies areas of increased risk incidence, based on the gained informational content – in financial transactions, user and account credentials;

- *hardware capabilities*, central to the identification of risks as the extent of resources determines the recovery times and error resistance; the budgetary constraints of the developers and users influence the future design and implementation of the components, as well as cryptographic tools in communication and authentication procedures;
- *software technologies*, deriving from the hardware capacity, as the efficiency and span of framework-implemented activities is dependent on the available hardware and projected activities; the extent of choices is determined by the deployment platform, communication capacity and environment owner as compared to the application owner;
- *component numbers*, directly controlling tolerance to incidents by allowing for activity autonomy and explicit redundancy as tools against overextended dependencies in components; there occurs a bidirectional effect on security, with increased component numbers allowing for better protection, yet increasing the occurrence probabilities for complexity-derived incidents;
- *channel numbers*, deriving from the number of components and influencing the risk susceptibility by exposing information and operational or analytical module endpoints to unauthorized interactions; the number of potential attacks depends on the number of inter-component messages, as it increases on chances to detect and impersonate authorized processes.

The criteria enumerated above, alongside factors ordered in Table 2, form the basis for the global assessor of security compliance, SC in the distributed application, by evaluating the effects and origination of incidents. The n factors in set $X = \{x_1, x_2, \dots, x_n\}$ are quantified by averaging their impact and including relative weight information, $w, w \in [0,1]$, as relating to a predefined system of measurement, where the comparison basis is formed by the optimum or total item number for the measured factor, represented by set $\overline{X} = \{\overline{x_1}, \overline{x_2}, \dots, \overline{x_n}\}$:

$$SC = \frac{\frac{\overline{x_1} - x_1}{\overline{x_1}} * (1 - w_1) \mid \frac{\overline{x_2} - x_2}{\overline{x_2}} * (1 - w_2) \mid \dots \mid \frac{\overline{x_n} - x_n}{\overline{x_n}} * (1 - w_n)}{w_1 + w_2 + \dots + w_n},$$

or in generic form as

$$SC = \frac{\sum_{i=1}^{n} \left(\frac{\overline{x_i} - x_i}{\overline{x_i}} * w_i \right)}{\sum_{i=1}^{n} w_i} = \frac{\sum_{i=1}^{n} \left[\left(1 - \frac{x_i}{\overline{x_i}} \right) * (1 - w_i) \right]}{\sum_{i=1}^{n} w_i},$$

where

x_i – factor *i* failure counter or costs;
$\overline{x_i}$ – total number of factor I items or value;
w_i – relative weight associated to factor $i, i = \overline{1, n}$.

In Table 2, the 8 levels of application impact, starting with credential discovery or loss and ending with subtle variations in distributed application behavior observed at various moments infer on the severity of the loss. For the two observed incidents in Table 3,

considering compatible systems and units of measurement, the weights reflect the order of the impacted factor's performance as follows – the first one, impacting MERICS.LOGICAL, is a fifth level risk source, and therefore is assigned a weight of 5/8 or 0.625. The second one, in WEBAPP and level 4, is slightly lower in impact and therefore corresponds to weight 4/8 = 0.5. MERICS contains only one instance of MERICS.LOGICAL, yet two autonomous, self replacing interfaces. Therefore, the global security compliance over the period covering these two incidents is calculated as

$$SC = \frac{\frac{1-1}{1} * (1 - 0{,}625) + \frac{2-1}{2} * (1 - 0{,}5)}{0{,}625 + 0{,}5} = 0{,}22$$

The 22% security compliance reflects the impending need for updating the targeted components, especially the logical module, whose failure is augmented by its uniqueness.

3. Metrics validation

Validation is a process which assures that a result complies with the expectations and the desired standards. While developing distributed computer applications it is necessary to find ways for quantifying the level of compliance. Numeric values are assigned to specific features of the informatics products. This is done using metrics. In measuring, validation tells weather the usage of a metric will lead to a satisfactory result or not. Both validation of the model and validation of the result are done.

Each model has its own metrics. Validation of the model is done by verifying the properties for each of the model's metrics. Generally a metric has the following representation [7]:

$y = f(X)$,

with $X = [x_1, x_2, ..., x_n]$ the set of influence factors for the informatics product characteristic which is being measured, as detailed in the previous section.

The properties to be verified in the model's validation process are:

- metric's sensitivity
- the non-catastrophic character
- the non-compensatory character

The model is sensitive when the variation of the influence factors produces the variation of the measured value: y. For a small variation of the influence factors values the variation of the resulted value is small and for a big variation of the influence factors values the variation of the resulted value is big[8]. A model $y = f(x)$ is sensitive when the following is true:

$$f(x + h) = f(x) + f(h)$$

The model has a catastrophic character when there are values of the influence factors for which the measured value y is impossible to calculate. This is the situation when dividing a number to a value which tends to be zero.

The model has a compensatory character when for different values of the influence factors the result is the same. It is important in developing computer programs that for different input the output is different as well. The above model has a non-compensatory character when the next condition is true:

$$f(x_1) \neq f(x_2) \; \forall \; x_1 \neq x_2$$

Let TD be a set of MERICS testing data used to validate t Security Compliance *SC* indicator.

$$TD = \{td_1, td_2, ..., td_{ntd}\}$$

Where:

ntd – the number of test data sets used;

$td_1, td_2, ..., td_{ntd}$ – the test data sets;

pv_{ij} – the property value for the test data set td_i.

For validating the model properties [7], the Table 4 is populated with values corresponding to each property of a test data set: sensitivity, non-catastrophic character, non-compensatory character. At the intersection of the line *i* with the column *j* the value of pv_{ij} is 1 when the *SC* property is verified. If the SC property is not verified the value of pv_i is 0.

Table 4 – Indicator properties validating [7]

Test Data Set\Indicator Property	Sensitive	Non-Catastrophic	Non-Compensatory
td_1	pv_{11}	pv_{12}	pv_{13}
td_2	pv_{21}	pv_{22}	pv_{23}
...	...	...	...
td_i	pv_{i1}	pv_{i2}	pv_{i3}
...	...	...	...
td_{ntd}	pv_{ntd1}	pv_{ntd2}	pv_{ntd3}
	TPV_1	TPV_2	TPV_3

Where:

td_i – the test data set *i* used as input in MERICS application for validating the *SC* indicator;

ntd – the total number of test data sets;

pv_{ij} – the propriety value as 0 or 1 indicating whether the property is verified or not;

TPV_i – total property value used to express the level of property verification by aggregation of pv_{ij}.

Knowing the above *ntd* and pv_{ij}, the aggregated property value, TPV_j is given by:

$$TPV_j = \frac{\sum_{i=1}^{ntd} pv_{ij}}{ntd}, \; j=1..3$$

Knowing the above TPV_j, the indicator I_{SC} is used to validate the security compliance model and is given by:

$$I_{SC} = \frac{\sum_{j=1}^{3} 0.33 * TPV_j}{ntd}$$

The value of I_{SC} gives the validation of *SC* as following:

- If the value of $I_{SC} < 0.78$ the *SC* indicator is not validated
- If the value of $I_{SC} \geq 0.78$ *and* $I_{SC} < 0.92$ the *SC* indicator is validated as *good*
- If the value of $I_{SC} \geq 0.92$ the *SC* indicator is validated as *very good*

The model is being refined using MERICS and it is to be verified with every version of the informatics application.

Conclusions

Developers and end-users communication-context threat awareness is required in order to provide the mechanisms of defense – disaster recovery documentation and procedural detailing, increased security in vulnerable sections, identified through the evaluation using automated modeling by analytical modules. The development and refining of risk assessment models and associated metrics enables the owners or developers of complex software applications to measure, quantify risk and evaluated individual and global behavior through successive stages of the application's lifecycle and associated versions of the assemblies and software structures that form the implemented content.

References

1. Benaroch, M. and Appari, A., **Financial Pricing of Software Development Risk Factors**, IEEE Software, Volume 27, Issue 5, October 2010, pp. 65 - 73, ISSN 0740-7459.
2. Judea Pearl, **Causality: Models, Reasoning and Inference**, 2nd Edition, Cambridge University Press, 2009, 484 pp, ISBN 978-0521895606.
3. Keshlaf, A.A. and Riddle, S., **Risk Management for Web and Distributed Software Development Projects**, 2010 Fifth International Conference on Internet Monitoring and Protection (ICIMP), 9-15 May 2010, pp. 22 - 28, ISBN 978-1-4244-6726-6.
4. Munch, J, **Risk Management in Global Software Development Projects: Challenges, Solutions, and Experience**, 2011 Sixth IEEE International Conference on Global Software Engineering Workshop (ICGSEW), 15-18 August 2011, pp. 35 - 35, ISBN 978-1-4577-1839-7.
5. Sadiq, M., Rahmani, M.K.I., Ahmad, M.W. and Sher Jung, **Software risk assessment and evaluation process (SRAEP) using model based approach, Networking and Information Technology (ICNIT)**, 2010 International Conference, 11-12 June 2010, pp. 171 – 177, ISBN 978-1-4244-7579-7.
6. Saleem, M.Q., Jaafar, J. and Hassan, M.F., **Model driven security frameworks for addressing security problems of Service Oriented Architecture**, 2010 International Symposium in Information Technology (ITSim), 15-17 June 2010, pp. 1341 – 1346, ISBN 978-1-4244-6715-0.
7. Ion Ivan, Catalin Boja, **Metode Statistice in analiza software**, Editura ASE, Bucuresti, 2004, 482 pg, ISBN 973-594-498-7
8. Adrian Mihai Vişoiu, **Rafinarea Metricilor Software**, PhD Thesis

[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. His concerns lie in the fields 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 concerning the development of commercial applications.

[2]Emanuel Eduard HERTELIU has graduated the Faculty of Economic Cybernetics, Statistics and Informatics, in 2009, as part of the Bucharest Academy of Economic Studies, followed by the Economic Informatics Master Program and is currently a PhD candidate as part of the same institution. He is interested in software development using computer programming languages as C++, C# and Java, as well as Web-oriented application programming, using Html, Php or the ADO.NET technology.