UNIVERSIDADE TÉCNICA DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

CEO Framework
Information System Architecture
Evaluation Metrics

André Ferreira Ferrão Couto e Vasconcelos

Technical Report – CEO, INESC
Version: 1.1

July 2007

CEO Framework Metrics

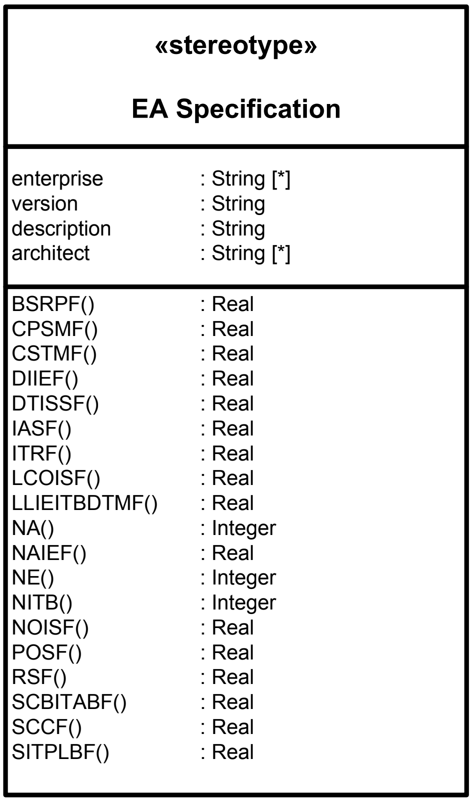


Figure 1 – Metrics proposed integrated in the CEOF UML profile

Table 1 – Auxiliary OCL methods for the metrics definition (at «EA Specification» level)

Method	Result
`AllG(): Set(Goal)`	`Set of «Goal» in an Architecture`
`AllP(): Set(Process)`	`Set of «Process» in an Architecture`
`AllR(): Set(Resource)`	`Set of «Resource» in an Architecture`
`AllB(): Set(Block)`	`Set of «Block» in an Architecture`
`AllS(): Set(Service)`	`Set of «Service» in an Architecture`
`AllITAB(): Set(IT_Application_Block)`	`Set of «IT Application Block» in an Architecture`
`AllISS(): Set(IS_Service)`	`Set of «IS Service» in an Architecture`
`AllBS(): Set(IS_Service)`	`Set of «Business Service» in an Architecture`
`AllIE(): Set(Information_Entity)`	`Set of «Information Entity» in an Architecture`
`AllIEO():`	`Set of «Information Entity» in an Architecture, without the`

Method	Result
Set(Information_Entity)	descendent classes
AllLLIE(): Set(Information_Entity)	Set of «Low Level Information Entity» in an Architecture
AllITPB(): Set(IT_Presentation_Block)	Set of «IT Presentation Block» in an Architecture
AllITLB(): Set(IT_Logic_Block)	Set of «IT Logic Block» in an Architecture
AllITB(): Set(IT_Block)	Set of «IT Block» in an Architecture
AllISB(): Set(IS_Block)	Set of «IS Block» in an Architecture
AllISO(): Set(IS_Operation)	Set of «IS Operation» in an Architecture
TOIE(): Integer	Total Number of «Information Entity» in an Architecture, not considering descendant classes (as «Low Level Information Entity»)
TROIE(): Integer	Total Number of relations between «Information Entity» in an architecture, not considering descendent classes (as «Low Level Information Entity»)
TP(): Integer	Total Number of «Process» in an Architecture
TISO(): Integer	Total Number of «IS Operation» in an Architecture
TISB(): Integer	Total Number of «IS Block» in an Architecture
TITAB(): Integer	Total Number of «IT Application Block» in an Architecture
TISS(): Integer	Total Number of «IS Service» in an Architecture
TITPB(): Integer	Total Number of «IT Presentation Block» in an Architecture
TITLB(): Integer	Total Number of «IT Logic Block» in an Architecture
TIE(): Integer	Total Number of «Information Entity» in an Architecture
TITB(): Integer	Total Number of «IT Block» in an Architecture
TBS(): Integer	Total Number of «Business Service» in an Architecture
TPOS(): Integer	Sum of the number of possible operating systems in each «IT Application Block»
TITSISS (): Integer	Sum of the number of "IT Service", which attribute serviceType is equal to "Integration Service", related to each "IS Service"
TLLIEIE (): Integer	Sum of the number of «Low Level Information Entity» that is related with each «Information Entity»
TLLIE (): Integer	Number of «Low Level Information Entity» in an Architecture
TSITPB (): Integer	Number of «IT Presentation Block» which attribute state value is *stateful*
TSITLB (): Integer	Number of «IT Logic Block» which attribute state value is *stateful*
TSITB (): Integer	Number of «IT Block» attribute which attribute securityElement value is true
SITAB(c: IT_Application_Block): Integer	Number of «IT Block» which attribute securityElement value is true between the «IT Application Block» provided and the remaining
TSITAB(): Integer	Total Number of security components existing in all «IT Block»
TIES(): Integer	Total Number of sets of «Information Entity» for all «IS Block»
TISOISB(): Integer	Sum of «IS Operation», per «IS Block»
countISBlock(s: Set(Classifier)): Integer	Number of elements of type «IS Block» in the "s" set
getAllClassifierInteraction (inter: Interaction): Set (Classifier)	Set of classifiers that are part of an interaction (e.g., sequence diagram,...)
getAllMessageInteraction (inter: Interaction): Set (Message)	Set of messages that are part of an interaction (e.g., sequence diagram,...)
getAllElems (): Set (Classifier)	All the Specification Elements
belongToSpec(c: Classifier): Boolean	Verifies if a classifier belongs to the architecture specification

Method	Result
getAllInteractions(): Set(Interaction)	Returns all the interactions that occurred using objects from the current specification
TMcCabe (): Integer	Total of McCabe metric applied to IS, for this architecture
TISBSS(): Integer	Total number of «IS Block» used, in each «IS Service» or «Business Service»
TPM(): Integer	Total number of «Process» that aren't supported in «IS Block» that support critical and non-critical processes
TISBIE (): Integer	Total number of «IS Block» that CUD (Create, Update or Delete) information entities, per information entity
TIESM(): Integer	Total number of «Information Entity» that are used (CUD) by «IS Block» that support «Information Entity» with and without security restrictions
TLLIEITBM(): Integer	Total number of «Low Level Information Entity» that are used (CUD) by «IT Block» that support «Low Level Information Entity» with primitive and derived data
TRBS(): Integer	Total number of «Business Service» required by each «process»
TRNIBS(): Integer	Total number of «Business Service» required by each «process» that weren't provided (or implemented in «IS Block»)
TRITB (): Integer	Total number of «IT Block» which attribute redundantElement is true
TISBITBM (): Integer	Total number of «IS Block» that are critical but are implemented in «IT Block» which support other «IS Block» non-critical and vice-versa

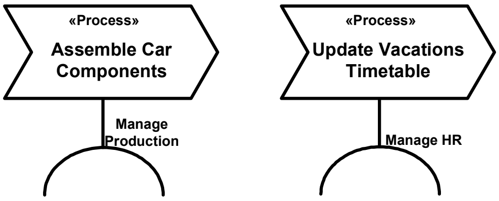
Table 2 – Metric description template

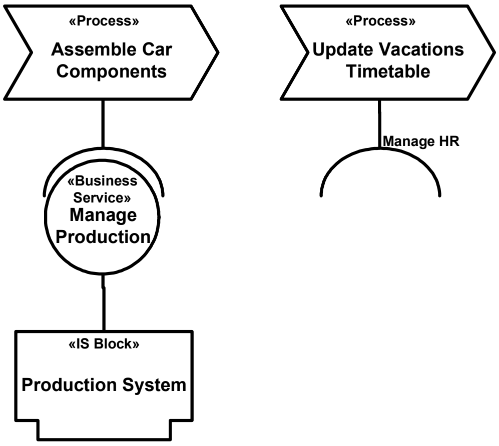
Acronym	Metric Acronym
Name	Metric Name
Computation	Description on the metric algorithm or formula
Formal Definition	OCL definition of the metric (using the auxiliary OCL methods described in Table 1)
Scale	Scale of possible values for the metric
Architectural Levels	Architecture levels relevant for this metric
ISA Primitives and attributes	Architectural primitives and attributes used in the metric computation
ISA Qualities	Enumeration of the IS quality characteristics related with the metric
Support	Rational that supports the metric proposed and its relevance for measuring the ISA qualities

Table 3 – Business Service Required and Provided Factor Metric

Acronym	*BSRPF*
Name	*Business Service Required and Provided Factor*
Computation	The Business Service Required and Provided Factor is computed considering the number of business services required and not provided by the IS.: $BSRPF = 1 - \frac{\sum_{i=1}^{\#«Bu\sin ess\ \Pr ocess»} \#«Bu\sin ess\ Service»RNI_i}{\sum_{i=1}^{\#«Bu\sin ess\ \Pr ocess»} \#«Bu\sin ess\ Service»R_i}$, where *#«Business Service»RNI$_i$* – is the number of *«Business Service»* Required for supporting process *i* and Not Implemented. *#«Business Service»R$_i$* – is the number of *«Business Service»* Required for supporting process *i*. *#«Business Process»* – is the number of Business Processes

Formal Definition	`context EA_Specification::BSRPF(): Real`
	`  pre: self.TRBS() > 0`
	`  post: result = 1- self.TRNIBS() / self.TRBS()`

Scale	[0; 1]
Arch. Levels	Business and Application Architecture
ISA Primitives and attributes	Primitives: *«Process»* ; *«Business Service»; «IS Block»*
ISA Qualities	The Suitability and Application/Business architecture alignment qualities tend to increase with this metric.
Support	This metric measures the alignment and the suitability of the services provided by the information systems to the business processes. This analysis is accomplished considering all the services required by processes that are not support by «business services» (or which «business services» are not implemented in any application component («IS Block») (Maes et al. 2000).
Example	ISAA  «Process» Assemble Car Components «Process» Update Vacations Timetable Manage Production Manage HR $\text{BSRPF} = 1 - \frac{1+1}{2} = 0$
	ISA B

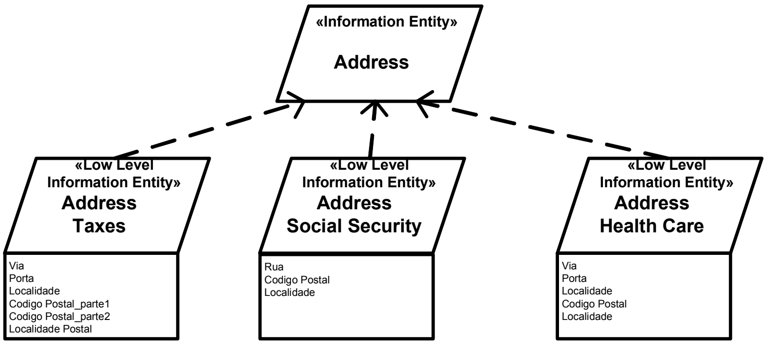


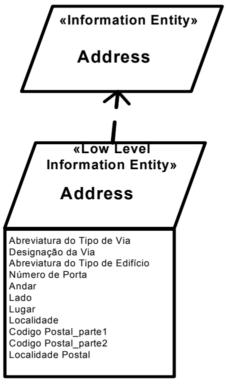
$$\text{BSRPF} = 1 - \frac{0+1}{2} = \frac{1}{2}$$

Table 4 – Different Implementations of Information Entity Factor Metric

Acronym	*DIIEF*
Name	*Different Implementations of Information Entity Factor*
Computation	The Different Implementations of Information Entity Factor is computed counting, for each *«Information Entity»* the number of possible implementations in *«Low Level Information Entity»* $DIIEF = \frac{\#\text{«Information Entity»}}{\sum_{i=1}^{\#\text{«Information Entity»}} NLLIE_i}$, where: *NLLIE$_i$* – is the number of «Low Level Information Entity» associated to the *«Information Entity»$_i$* thought the «implements» relation *#«Information Entity»* – is the total number of *«Information Entity»* in the ISA

Formal Definition	`context EA_Specification::DIIEF(): Real`
	`  pre: self.TLLIEIE() > 0`
	`  pre: self.TIE() >= self.TLLIEIE()`
	`  post: result = self.TIE() / self.TLLIEIE()`

Scale	]0; 1]
Arch. Levels	Information Architecture
ISA Primitives and attributes	Primitives: *«Low Level Information Entity»* ; *«Information Entity»*
ISA Qualities	The Semantic interoperability tends to increase with this metric.
Support	This metric measures the number of different implementations that exist for each information entity. According to Inmon (2000), for each information entity ("top level") there might be other entities that implement it ("low level information entity"). The existence of different «Low Level Information Entities» points to semantic problems for that «Information Entity» (e.g., by using different formats or attributes in the implementation of the information entity).
Example	ISA A  «Information Entity» Address «Low Level Information Entity» Address Taxes Via Porta Localidade Codigo Postal_parte1 Codigo Postal_parte2 Localidade Postal «Low Level Information Entity» Address Social Security Rua Codigo Postal Localidade «Low Level Information Entity» Address Health Care Via Porta Localidade Codigo Postal Localidade $DIIEF = \frac{1}{3}$
	ISA B

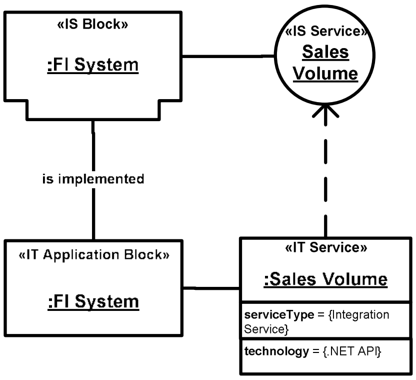


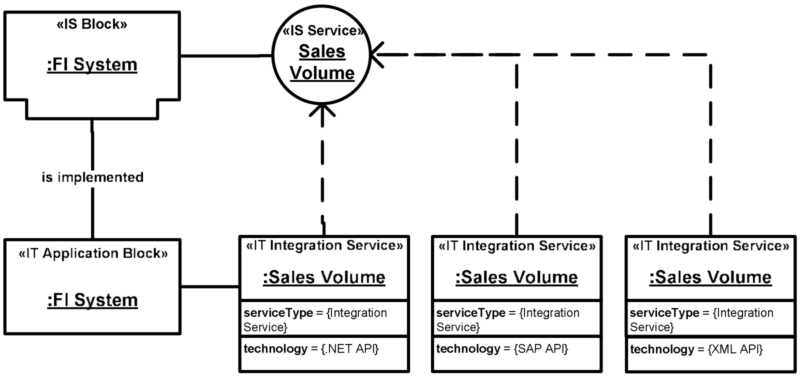
$$DIIEF = \frac{1}{1} = 1$$

Table 5 – Distinct Technologies for IS Services Factor Metric

Acronym	*DTISSF*
Name	*Distinct Technologies for IS Services Factor*
Computation	The Distinct Technologies for IS Services Factor is computed counting for each «IS Service» the number of «IT Service» of type "*Integration Service*". $DTISSF = 1 - \frac{\#«IS\ Service»}{\sum_{i=1}^{\#«IS\ Service»} \#«IT\ Service»_{Integration\ \ i}}$, where: *#«IT Service» Integration i* – is the number of *«IT Service»*, which attribute *serviceType* is equal to *"Integration Service"* that implements the *«IS Service» i* *#«IS Service»* – is the number of *«IS Service»* in the ISA

Formal Definition	`context EA_Specification::DTISSF(): Real`
	`  pre: self.TITSISS() > 0`
	`  pre: self.TISS() >= self.TITSISS()`
	`  post: result = 1 - self.TISS() / self.TITSISS()`

Scale	[0; 1[
Arch. Levels	Application and Technology Architectures
ISA Primitives and attributes	Primitive: *«IT Service»*; Attribute: *serviceType*
ISA Qualities	The technical interoperability and the portability of an IS tend to increase with this metric.
Support	The technical interoperability of a software architecture increases by providing the same interface in different technologies (Sarkis and Sundarraj 2003, section 3.2.1). In the same way, with this metric the technical interoperability and portability of an EIS is analyzed as the average of the Technologies that each application interface provides.
Example	ISA A  «IS Block» :FI System «IS Service» Sales Volume is implemented «IT Application Block» :FI System «IT Service» :Sales Volume serviceType = {Integration Service} technology = {.NET API} $DITSSF = 1 - \frac{1}{1} = 0$
	ISA B



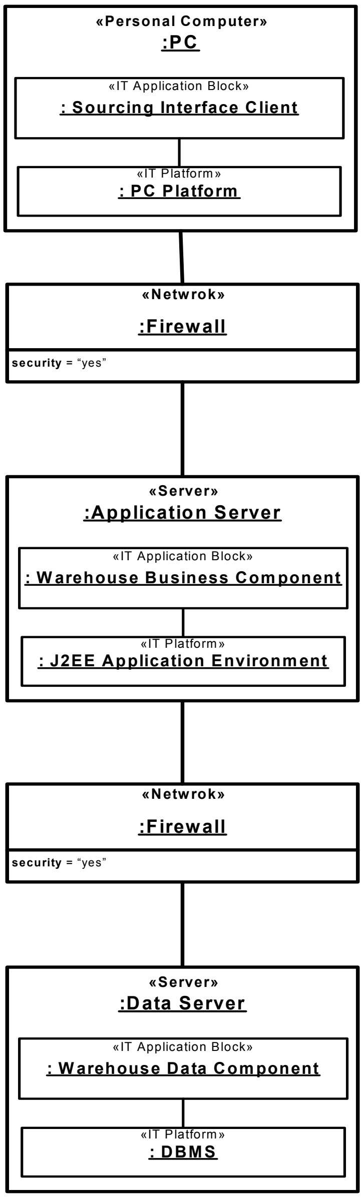
$$DITSSF = 1 - \frac{1}{3} = \frac{2}{3}$$

Table 6 – Security Components Between «IT Application Block» Factor Metric

Acronym	*SCBITABF*
Name	*Security Components Between «IT Application Block» Factor*
Computation	The Security Components Between «IT Application Block» Factor is computed counting, for each *«IT Application Block»* the minimal number of *«IT Block»,* which attribute "securityElement" is *true*, that is between the path of that block and each of the remaining *«IT Application Block»*. $SCBITABF = \frac{\sum_{i=1}^{\#IT\ Application\ Block} \left[\sum_{j=1}^{\#\text{«}IT\ Application\ Block\text{»}} Min\ \{\# SITB_{ij}\} \right]}{\# \text{«}IT\ Application\ Block\text{»} \times \# \text{«}ITBlock\text{»}}$, where: *Min{#SITB$_{ij}$}* – is the minimal number of instances of *«IT Block»,* which attribute "securityElement" has the value "true", and is in the path between *«IT Application Block»$_i$* and *«IT Application Block»$_j$.* *#«IT Application Block»* – is the number of instances of *«IT Application Block»* *#«IT Block»* – is the number of instances of *«IT Block»*

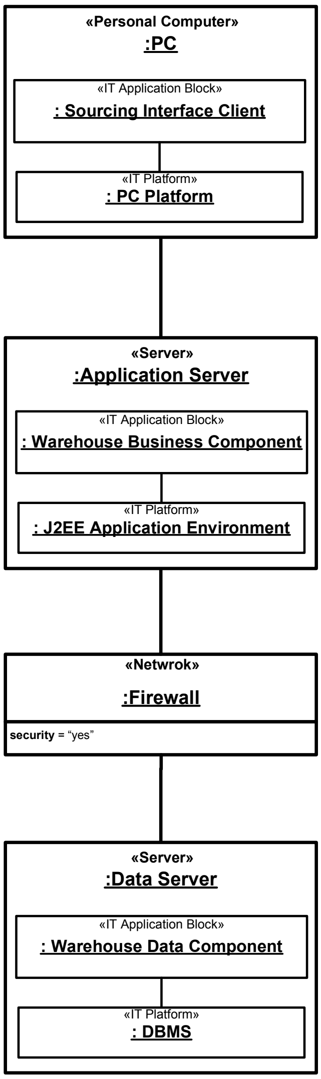
Formal Definition	`context EA_Specification::SCBITABF(): Real`
	`  pre: self.TITAB() > 0`
	`  post: result = self.TSITAB() / (self.TITAB() * self.TITB())`

Scale	[0; 1]
Arch. Levels	Technology Architecture
ISA Primitives and attributes	Primitives: *«IT Block», «IT Application Block»*; Attribute: *security*
ISA Qualities	The security of an ISA tends to increase with this metric.
Support	The ISA security is increased by putting security elements on it, as IDS, firewalls, etc. Thus, this metric, is not limited to counting the number of security components but it also considers, for each application component, the number of security components that isolate it from other components.
Example	ISA A ISA B



$$NSCBITAB = \frac{(1+2)+(1+1)+(1+2)}{3x11}$$

$$NSCBITAB = \frac{8}{33} \approx 24\%$$



$$NSCBITAB = \frac{(1+1)+(0+1)+(0+1)}{3x10}$$

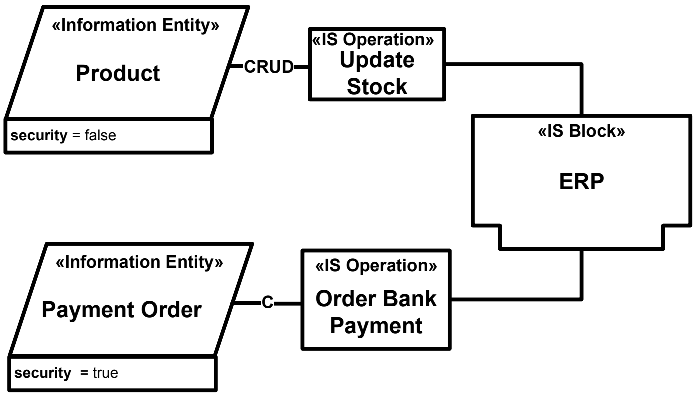
$$NSCBITAB = \frac{4}{30} \approx 13\%$$

Table 7 – Information-Application Security Factor Metric

Acronym	*IASF*
Name	*Information-Application Security Factor*
Computation	The Information-Application Security Factor is computed considering the number of information entities with high level security requirements supported in «IS Blocks» that also support information entities without high security requirements and vice versa. $IASF = 1 - \frac{\#\{InformationEntity_S \in ISBlock_{NS}\} + \#\{InformationEntity_{NS} \in ISBlock_S\}}{\#«InformationEntity»}$, where: $\#\{InformationEntity_S \in ISBlock_{NS}\}$ – is the number of «Information Entities» that its Security attribute value is {Yes} supported in «IS Blocks» that support other «Information Entities» which Security attribute value is {No}; where an «Information Entity» is “supported” by an «IS Block» if and only if exists at least one «operation» provided by the «IS Block» that CUD the «Information Entity». $\#\{InformationEntity_{NS} \in ISBlock_S\}$ – is the number of «Information Entities» that its Security attribute value is {No} supported in «IS Blocks» that support other «Information Entities» which Security attribute value is {Yes}; where an «Information Entity» is “supported” by an «IS Block» if and only if exists at least one «operation» provided by the «IS Block» that CUD the «Information Entity». *#«Information Entity»* – is the number of information entities

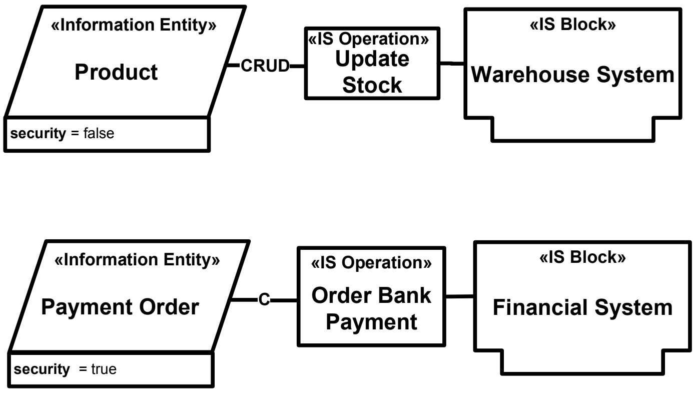
Formal Definition	`context EA_Specification::IASF(): Real`
	`  pre: self.TOIE() > 0`
	`  post: result = 1-self.TIESM() / self.TOIE()`

Scale	[0; 1]
Arch. Levels	Information and Application Architectures
ISA Primitives and attributes	Primitives: *«IS Block»* , *«ISOperation», «Information Entity»;* Attribute: *Security*
ISA Qualities	The Security and the Information-Application Architectures alignment tend to increase with this metric.
Support	According to Sousa, Pereira and Marques (2004) applications should manage information entities of the same security level.
Example	ISA A



$$IASF = 1 - \frac{1+1}{2} = 0$$

ISA B



$$IASF = 1 - \frac{0+0}{2} = 1$$

Table 8 – IT Redundancy Factor Metric

Acronym	*ITRF*
Name	*IT Redundancy Factor*
Computation	The IT Redundancy Factor is computed counting the *«IT Block»* which attribute "*redundantElement*" is true $ITRF = \frac{\# RITB}{\# «IT\ Block»}$, where: *#SITB* – is the number of *«IT Block»* which attribute "redundantElement" has the value true *#«IT Block»* – is the number of *«IT Block»*

Formal Definition	`context EA_Specification::ITRF(): Real`
	`  pre: self.TITB() > 0`
	`  post: result = self.TRITB() / self.TITB()`

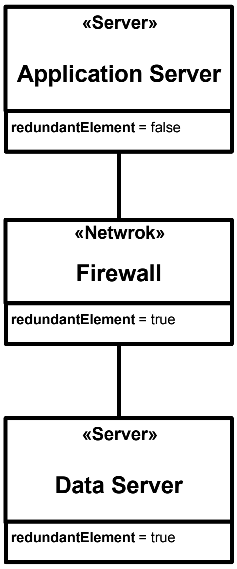
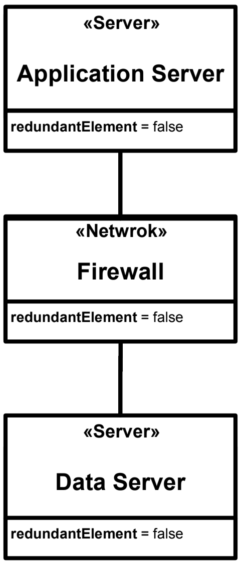
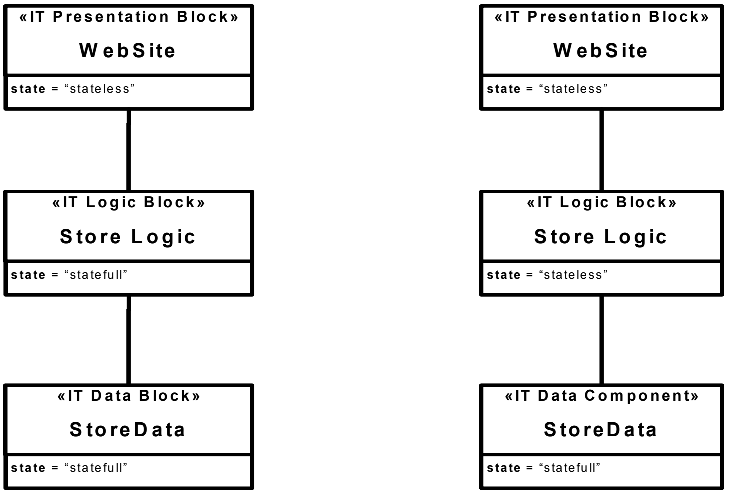
Scale	[0; 1]
Arch. Levels	Technology Architecture
ISA Primitives and attributes	Primitive: *«IT Block»*, Attribute: *redundantElement*
ISA Qualities	The Fault tolerance tends to increase with this metric.
Support	The Fault tolerance of an ISA tends to increase by using redundant elements (Vargas 2000). This metrics echoes this fact.
Example	ISA A  «Server» Application Server redundantElement = false «Netwrok» Firewall redundantElement = true «Server» Data Server redundantElement = true $ITRF = \frac{2}{3}$ ISA B  «Server» Application Server redundantElement = false «Netwrok» Firewall redundantElement = false «Server» Data Server redundantElement = false $ITRF = \frac{0}{3} = 0$

Table 9 – Stateful «IT Presentation Block» and «IT Logic Block» Factor Metric

Acronym	*SITPLBF*
Name	*Stateful «IT Presentation Block» and «IT Logic Block» Factor*
Computation	The Stateful «IT Presentation Block» and «IT Logic Block» Factor is computed counting the number of «IT Presentation Block» and «IT Logic Block» that its attribute "state" value is "stateful" $\text{SITPLBF} = 1 - \frac{\#SITPLB}{\#«IT\ \text{Pr}\,esentationBlock» + \#«IT\ LogicBlock»}$, where: *#SITPLB* – is the number of *«IT Presentation Block»* and *«IT Logic Block»* that its attribute "state" value is "stateful". *#«IT Presentation Block»* – is the number of *« IT Presentation Block »* *#«IT Logic Block»* – is the number of *« IT Logic Block »*

Formal Definition	`context EA_Specification::SITPLBF(): Real`
	`  pre: self.TITLB() + self.TITPB() > 0`
	`  post: result = 1- (self.TSITLB() + self.TSITPB()) / (self.TITLB()`
	`                                                    + self.TITPB())`

Scale	[0; 1]
Arch. Levels	Technology Architecture
ISA Primitives and attributes	Primitives: *«IT Presentation Block», «IT Logic Block», «IT Data Block», «IT Coordination Block»*; Attribute: *state*
ISA Qualities	The scalability of an ISA tends to increase with this metric.
Support	The Scalability of an EIS is increased if business and presentation components do not keep the state (since it will be easier for implementing new parallel instances of these ISA components) – BEA (2006). The Scalability of an ISA tend to grow if the «IT Presentation Blocks» and the «IT Logic Blocks» do not preserve the application state (stateless) – the «IT Data Blocks» should be the ones to keep application state.
Example	ISA A ISA B

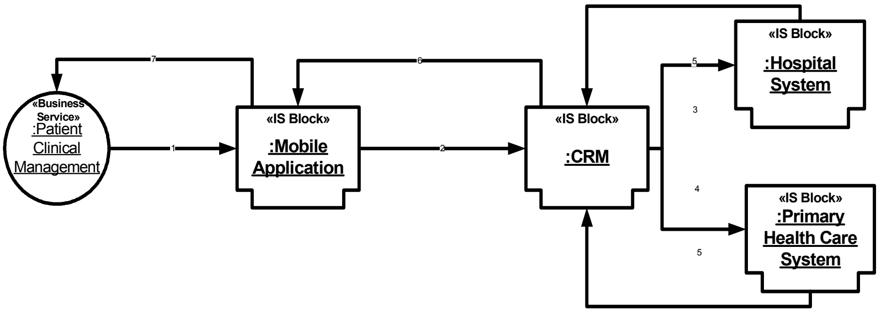


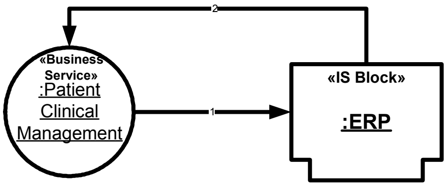
$$\text{SITPLBF} = 1 - \frac{1}{1+1} = \frac{1}{2} \qquad \text{SITPLBF} = 1 - \frac{0}{1+1} = 1$$

Table 10 – Service Cyclomatic Complexity Factor Metric

Acronym	*SCCF*
Name	*Service Cyclomatic Complexity Factor*
Computation	The Service Cyclomatic Complexity Factor is computed considering the number of dependencies between «IS Blocks» subtracted by the number of «IS Blocks» that support the service, for each service. $SCCF = \frac{\#«Bu\sin ess\ Service»+\#«IS\ Service»}{\sum_{i=1}^{\#«Bu\sin ess\ Service»+\#«IS\ Service»} \lvert e_i - n_i + 2 \rvert}$, where: e_i – is the number of dependencies between «IS Block» for the service *i.* n_i – is the number of «IS Blocks» that support the service *i.* #«Business Service» – is the number of «*Business Services*» #«IS Service» – is the number of «*IS Services*»

Formal Definition	`context EA_Specification::SCCF(): Real`
	`  pre: self.TISS()+ self.TBS() > 0`
	`  post: result = (self.TISS()+ self.TBS()) / (self.TMcCabe())`

Scale	]0; 1]
Arch. Levels	Application Architecture
ISA Primitives and attributes	Primitives: *«IS Block»* ; *«Business Service»; «IS Service»*
ISA Qualities	The Analyzability of an ISA tends to increase with this metric.
Support	Like McCabe (1976), for the software engineering area, considering that the higher the number of paths in a program, the higher its control flow complexity probably will be, in Vasconcelos et. al. (2005) is proposed a similar metric for evaluate the complexity of an ISA in the support of the business services – considering that the complexity, for each service, is measure by the difference between the number of dependencies and applications involved
Example	ISA A  «Business Service» :Patient Clinical Management «IS Block» :Mobile Application «IS Block» :CRM «IS Block» :Hospital System «IS Block» :Primary Health Care System 1 2 3 4 5 5 6 7 $SCCF = \frac{1}{8-4+2} = \frac{1}{6}$
	ISA B

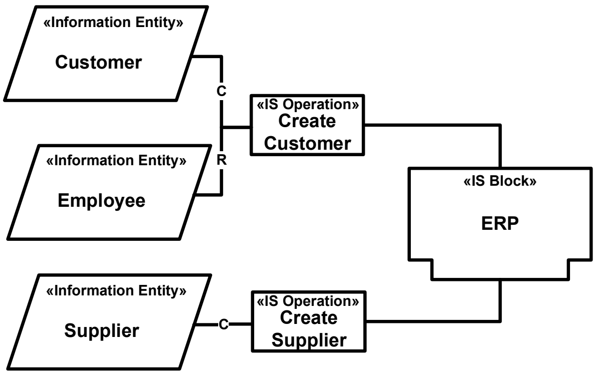


$$SCCF = \frac{1}{2-1+2} = \frac{1}{3}$$

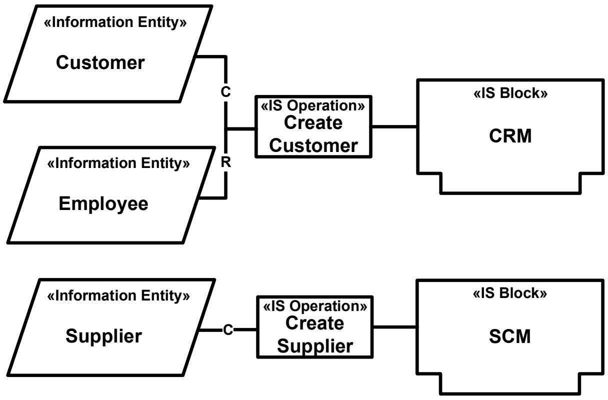
Table 11 – Lack of COhesion in «IS Block» Factor Metric

Acronym	*LCOISF*
Name	Lack of COhesion in «IS Block» Factor
Computation	The Lack of COhesion in «IS Block» Factor is computed counting the number of sets of information entities that are used by distinct functionalities of the same application (provided by operations in «IS Blocks»). $LCOISF = 1 - \frac{\sum_{i=1}^{\#IS\ Block} \#LCOIS_i}{\#«IS\ Block» \times \#«IS\ Operation» \times \#«Information\ Entity»}$, where: *#LCOIS*$_i$– is the number of sets of *«Information Entities» that are used by* «*operations*» distinct of the *«IS Block» i;.* *#«IS Block»* – is the number of *«IS Blocks»* *#«IS Operation»* – is the number of *«IS Operation»* *#«Information Entity»* – is the number of *« Information Entity»*

Formal Definition	`context EA_Specification::LCOISF(): Real`
	`  pre: self.TIE() > 0`
	`  pre: self.TISO() > 0`
	`  pre: self.TISB() > 0`
	`  post: result = 1 - self.TIES() / (self.TISB() * self.TISO() *`
	`                                                     self.TIE() )`

Scale	[0; 1]
Arch. Levels	Information and Application Architectures
ISA Primitives and attributes	Primitives: *«IS Block», «IS Operation», «Information Entity»*
ISA Qualities	The Changeability of an ISA tends to increase with this metric.
Support	This metric measure the correlation between application blocks and the information entities used in that application block. It is quantified by the average of the number of sets of information entities that are used by distinct operations of the same application. Vasconcelos et. al. (2005).
Example	ISA A  «Information Entity» Customer C «IS Operation» Create Customer «Information Entity» Employee R «IS Block» ERP «Information Entity» Supplier C «IS Operation» Create Supplier $LCOISF = 1 - \frac{1+1}{1 \times 2 \times 3} = \frac{2}{3}$

ISA B



$$LCOIS = 1 - \frac{1+1}{2 \times 2 \times 3} = \frac{5}{6}$$

Table 12 – Number of Operations in «IS Block» Factor Metric

Acronym	*NOISF*
Name	*Number of Operations in «IS Block» Factor*
Computation	The Number of Operations in «IS Block» Factor is computed counting the number of operations on each «IS Block» divided be the number of «IS Blocks». $NOISF = \frac{\#«IS\ Block»}{\sum_{i=1}^{\#IS\ Block} \#«IS\ operation»_{«ISBlock»i}}$, where: *#«operation»*$_{«IS\ Block»i}$ – is the number of operations on *«IS Block» i.* *#«IS Block»* – is the number of *«IS Block»*

Formal Definition	`context EA_Specification::NOISF(): Real`
	`  pre: self.TISO() > 0`
	`  pre: self.TISO()> self.TISB()`
	`  post: result = 1 - self.TISO() / self.TISB()`

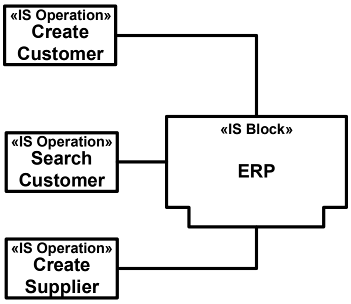
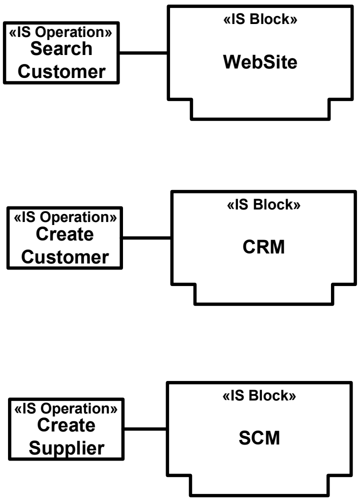
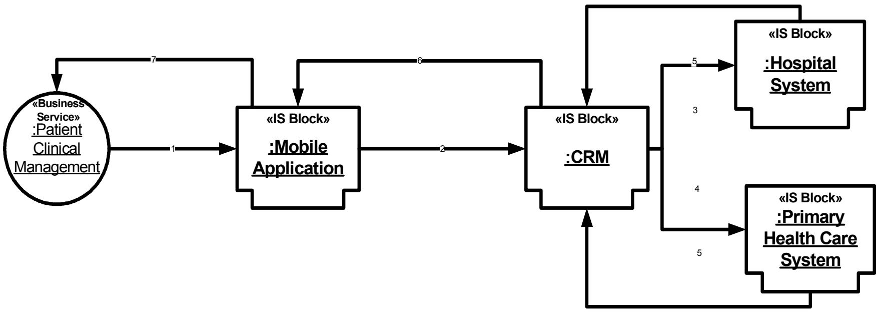
Scale	]0; 1]
Arch. Levels	Application architecture
ISA Primitives and attributes	Primitives: *«IS Block»* ; *«IS Operation»*
ISA Qualities	The adaptability and changeability (at application level) of an ISA tend to increase with this metric.
Support	The simplicity to adapt/alter operations in an ISA to new business demands is maximized when the impact of changing each operation is reduced to a certain application block («IS Block»). This metric measures this fact. This metric was defined considering the similar software engineering metric "Average number of methods per class", that considers the existing methods in each class (Abreu et al. 2004).
Example	ISA A  «IS Operation» Create Customer «IS Operation» Search Customer «IS Operation» Create Supplier «IS Block» ERP $NOISF = \frac{1}{3}$ ISA B  «IS Operation» Search Customer «IS Block» WebSite «IS Operation» Create Customer «IS Block» CRM «IS Operation» Create Supplier «IS Block» SCM $NOISF = \frac{3}{3} = 1$

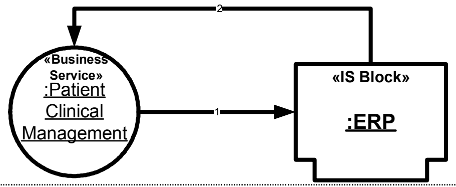
Table 13 – Response for a Service Factor Metric

Acronym	*RSF*
Name	*Response for a Service Factor*
Computation	The Response for a Service Factor is computed by considering the average of the number of «IS Blocks» that might be used to support each «Service». $RSF_{ISA} = \frac{\#«Bu\sin ess\ Service»+\#«IS\ Service»}{\sum_{i=1}^{\#«Bu\sin ess\ Service»+\#«IS\ Service»}\#«IS\ Block»_i}$, where: *#«IS Block»$_i$* – is the number of *«IS Blocks»* involved in supporting service *i.* *#«Business Service»* – is the number of «Business Services» *#«IS Service»* – is the number of *«IS Services»*

Formal Definition	`context EA_Specification::RSF(): Real`
	`  pre: self.TISS()+ self.TBS() > 0`
	`  post: result = (self.TISS()+ self.TBS()) / (self.TISBSS())`

Scale	]0; 1]
Arch. Levels	Application Architecture
ISA Primitives and attributes	Primitives: *«IS Block»* ; *«Business Service»*
ISA Qualities	The Testability of an ISA tends to increase with this metric
Support	This metric is similar to the software metric "Response For a Class" – see Chidamber and Kemerer (1994) and Basili (1996) for further details – that computes the number of methods that can potentially be executed in response to a message received. In Vasconcelos et. al. (2005) this metric is proposed (Average Response for a Service) and it computes the number of «IS Blocks» that might be used to support a service. In recent researches Sousa, Pereira and Marques (2004) suggest that each business process should be supported by the less number of applications as possible – this is also measure by this metric.
Example	ISA A  «Business Service» :Patient Clinical Management «IS Block» :Mobile Application «IS Block» :CRM «IS Block» :Hospital System «IS Block» :Primary Health Care System 1 2 3 4 5 5 6 7 $RSF = \frac{1}{4}$

ISA B



$$RSF = \frac{1}{1} = 1$$

Table 14 – Possible Operating Systems Factor Metric

Acronym	*POSF*
Name	*Possible Operating Systems Factor*
Computation	The Possible Operating Systems Factor is computed by counting, on each application («IT Application Block»), the number of possible operating systems (families) $POSF = 1 - \frac{\#\text{«}IT\ Application\ Block\text{»}}{\sum_{i=1}^{\#\text{«}IT\ Application\ Block\text{»}} NPOS_i}$, where: *$NPOS_i$* – is the number of possible operating systems families that the *«IT Application Block»$_i$* supports *#«IT Application Block»* – is the number of *«IT Application Block»* in the ISA

Formal Definition	`context EA_Specification::POSF(): Real`
	`  pre: self.TPOS() > 0`
	`  pre: self.TITAB() > self.TPOS()`
	`  post: result = 1 - self.TITAB() / self.TPOS()`

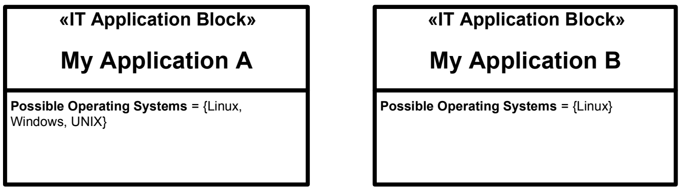
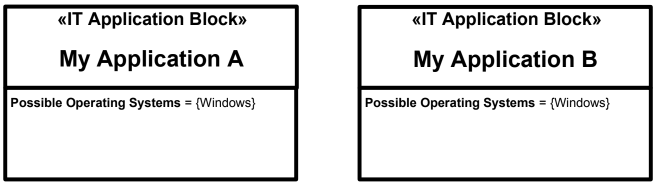
Scale	[0; 1[
Arch. Levels	Technology Architecture
ISA Primitives and attributes	Primitive: *«IT Application Block»*; Attribute: *possibleOperatingSystems*
ISA Qualities	The Portability (Adaptability) of an ISA tends to increase with this metric.
Support	The portability and Technical Interoperability in an ISA increase with the number of possible platforms where ISA components are able to operate (Sarkis and Sundarraj 2003, section 3.2.1). From a software engineering perspective, the portability of an operating system is a major indicator on an application portability; in the same way, the technical portability of an IS, represented by an ISA, is measure by this metric as the average of the software applications' («IT Application Block»)
Example	ISA A  «IT Application Block» My Application A Possible Operating Systems = {Linux, Windows, UNIX} «IT Application Block» My Application B Possible Operating Systems = {Linux} $POSF = 1 - \frac{2}{3+1} = \frac{1}{2}$ ISA B  «IT Application Block» My Application A Possible Operating Systems = {Windows} «IT Application Block» My Application B Possible Operating Systems = {Windows} $POSF = 1 - \frac{2}{1+1} = 0$

Table 15 – Critical Process - System Mismatch Factor Metric

Acronym	*CPSMF*
Name	Critical Process - System *Mismatch Factor*
Computation	The Critical Process - System Mismatch is computed by counting the number of critical business processes supported by «IS Blocks» that also support non-critical business processes and the number of non-critical business processes supported by «IS Blocks» that also support critical business processes $CPSMF = 1 - \frac{\#\{\Pr ocess_C \in ISBlock_{NC}\} + \#\{\Pr ocess_{NC} \in ISBlock_C\}}{\#«\Pr ocess»}$, where: $\#\{\Pr ocess_C \in ISBlock_{NC}\}$ – is the number of critical processes supported by *«IS Blocks»* that support other non-critical processes $\#\{\Pr ocess_{NC} \in ISBlock_C\}$ – is the number of non-critical processes supported by *«IS Blocks»* that support other critical processes *#«Process»* – is the number of processes

Formal Definition	`context EA_Specification::CPSMF(): Real`
	`  pre: self.TP() > 0`
	`  post: result = 1 - self.TPM() / self.TP()`

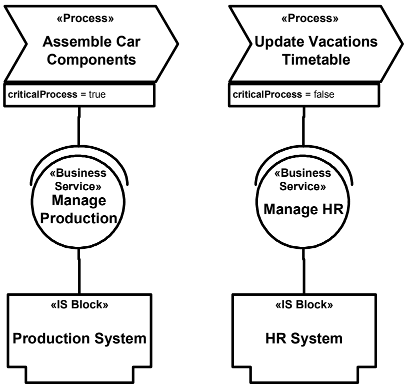
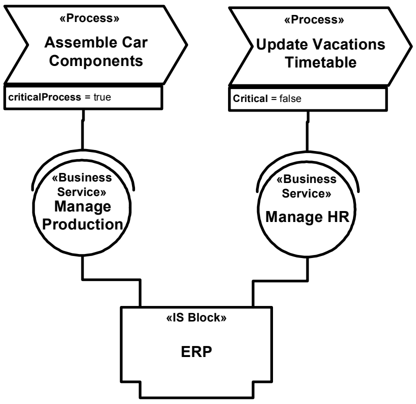
Scale	[0; 1]
Arch. Levels	Business and Application Architecture
ISA Primitives and attributes	Primitives: *«IS Block»* ; *«Process» - Attribute: criticalProcess*
ISA Qualities	The Business/Application Alignment of an ISA tends to increase with this metric..
Support	As described in Sousa, Pereira and Marques (2004) the critical business processes should be supported by different applications than non-critical business processes
Example	ISA A  «Process» Assemble Car Components criticalProcess = true «Process» Update Vacations Timetable criticalProcess = false «Business Service» Manage Production «Business Service» Manage HR «IS Block» Production System «IS Block» HR System $CPSMF = 1 - \frac{0+0}{2} = 1$ ISA B  «Process» Assemble Car Components criticalProcess = true «Process» Update Vacations Timetable Critical = false «Business Service» Manage Production «Business Service» Manage HR «IS Block» ERP $CPSMF = 1 - \frac{1+1}{2} = 0$

Table 16 – Number of Applications for «Information Entity» Factor Metric

Acronym	*NAIEF*
Name	*Number of Applications for «Information Entity» Factor*
Computation	The Number of Applications for «Information Entity» Factor » is computed counting the average number of applications («IS Blocks») that through its «operations» support each «information entity». $NAIEF = \dfrac{\#\text{«}Information\ Entity\text{»}}{\sum_{i=1}^{\#Information\,Entity}\left[\#\,ISBlocks \in \{\exists\text{«}ISOperation\text{»}\ CUD\ \text{«}InformationEntity\text{»}_i\}\right]}$, where: $\#\,ISBlocks \in \{\exists\text{«}operation\text{»}\ CUD\ \text{«}InformationEntity\text{»}_i\}$ – is the number of *«IS Blocks»* in which exists an «*operation*» that *CUD* (Creates, Updates or Deletes) the «information entity» *i.* *#«Information Entity» –* is the number of «*Information Entities*»

Formal Definition	`context EA_Specification::NAIEF(): Real`
	`  pre: self.TOIE() > 0`
	`  pre: self.TISBIE() > 0`
	`  pre: self.TOIE() >= self.TISBIE()`
	`  post: result = self.TOIE() / self.TISBIE()`

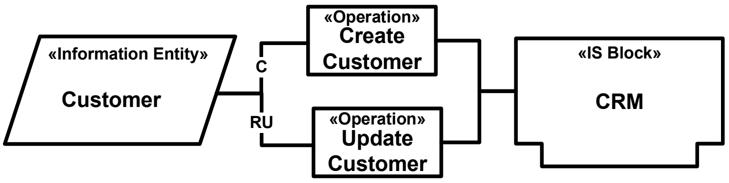
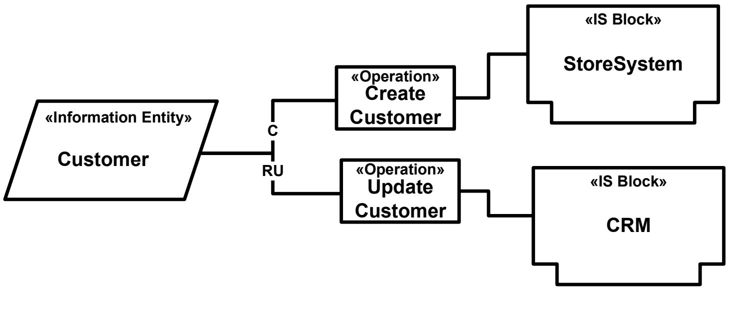
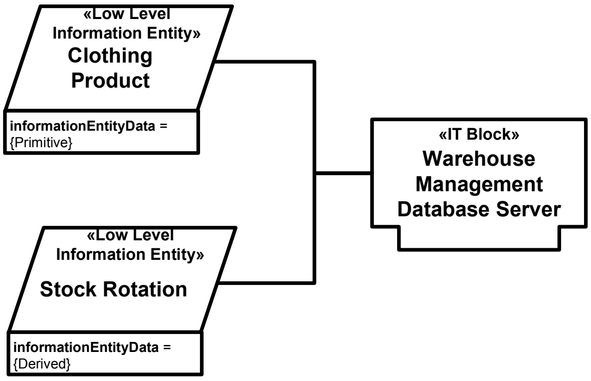
Scale	]0; 1]
Arch. Levels	Information and Application Architectures
ISA Primitives and attributes	Primitives: *«IS Block»*, *«Information Entity»*, *«IS Operation»*
ISA Qualities	The Information/application Alignment of an ISA tends to increase with this metric.
Support	According to Sousa, Pereira e Marques (2004) each information entity should be managed by a single application.
Example	ISA A  «Information Entity» Customer C RU «Operation» Create Customer «Operation» Update Customer «IS Block» CRM $NAIEF = \frac{1}{1} = 1$ ISA B  «IS Block» StoreSystem «Information Entity» Customer C RU «Operation» Create Customer «Operation» Update Customer «IS Block» CRM $NAIEF = \frac{1}{2}$

Table 17 – Low Level Information Entity – IT Block Data Type Mismatch Factor Metric

Acronym	*LLIEITBDTMF*
Name	*Low Level Information Entity – IT Block Data Type Mismatch Factor*
Computation	The Low Level Information Entity – IT Block Data Type Mismatch Factor is computed counting the number of «Low Level Information Entity» that are "primitive" that (according to the attribute "informationEntityData") supported in «IT Block» that also support «Low Level Information Entity» that are "derived" and vice versa. $\text{LLIEITBDTMF} = 1 - \frac{\#\{LowLevelInformationEntity_P \in ITBlock_{NP}\}}{\#«LowLevelInformationEntity»} + + \frac{\#\{LowLevelInformationEntity_D \in ITBlock_{ND}\}}{\#«LowLevelInformationEntity»}$, where: $\#\{LowLevelInformationEntity_P \in ITBlock_{NP}\}$ – is the number of *«Low Level Information Entity» which informationEntityData attribute is* {*Primitive*} supported in *«IT Block»* that support other *«Low Level Information Entity»* which attribute *informationEntityData is different from* {*Primitive*}. $\#\{LowLevelInformationEntity_D \in ITBlock_{ND}\}$ – is the number of *«Low Level Information Entity» which informationEntityData attribute is* {*Derived*} supported in *«IT Block»* that support other *«Low Level Information Entity»* which attribute *informationEntityData is different from* {*Derived*}. *#«Low Level Information Entity»* – number of low level information entities

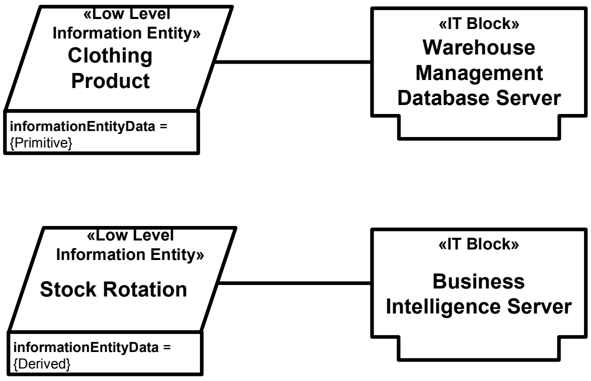
Formal Definition	`context EA_Specification::LLIEITBDTMF(): Real`
	`  pre: self.TLLIE() > 0`
	`  post: result = 1 - self.TLLIEITBM() / self.TLLIE()`

Scale	[0; 1]
Arch. Levels	Information and Technology Architecture
ISA Primitives and attributes	Primitive: *«IT Block»; «ISOperation»*;*«Low Level Information Entity»;* Attribute: *informationEntityData*
ISA Qualities	The Information/technology Alignment of an ISA tends to increase with this metric.
Support	According to Inmon the primitive and derived data present important differences on performance, accessing patterns, availability, among others issues (Inmon 1992). Thus, is consider a "good architectural practice" using different technology components to support primitive and derived data – using this metric it is possible to measure the alignment between the technology and the information architectures.
Example	ISA A


«Low Level Information Entity» Clothing Product
informationEntityData = {Primitive}
«Low Level Information Entity» Stock Rotation
informationEntityData = {Derived}
«IT Block» Warehouse Management Database Server

$$\text{LLIEITBDTM F} = 1 - \frac{1+1}{2} = 0$$

ISA B


«Low Level Information Entity» Clothing Product
informationEntityData = {Primitive}
«IT Block» Warehouse Management Database Server
«Low Level Information Entity» Stock Rotation
informationEntityData = {Derived}
«IT Block» Business Intelligence Server

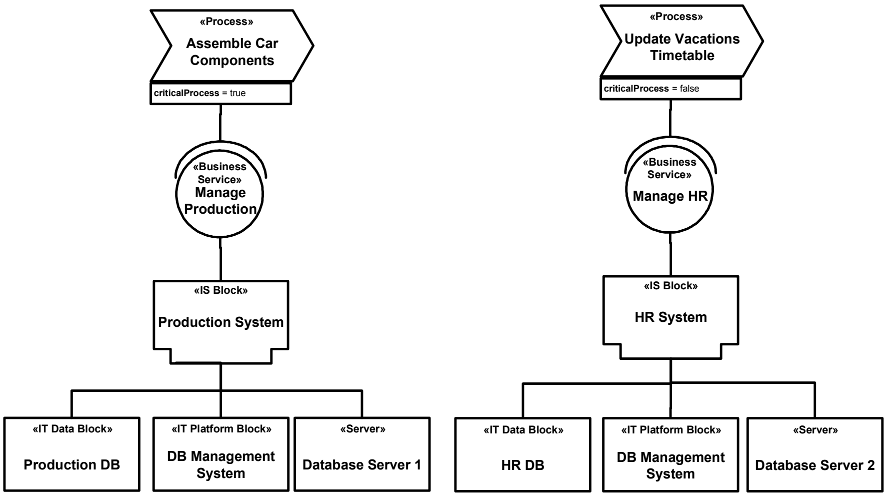
$$LLIEITBDTMF = 1 - \frac{0+0}{2} = 1$$

Table 18 – Critical System - Technology Mismatch Factor Metric

Acronym	*CSTMF*
Name	*Critical System - Technology Mismatch Factor*
Computation	The Critical System - Technology Mismatch Factor is computed counting the number of «IS Block» considered critical (for supporting exclusively critical processes) supported in «IT Block» that support «IS Block» not critical and vice versa. $CSTMF = 1 - \frac{\#\{ISBlock_C \in ITBlock_{NC}\} + \#\{ISBlock_{NC} \in ITBlock_C\}}{\#«ISBlock»}$, where: $\#\{ISBlock_C \in ITBlock_{NC}\}$ – is the number of «ISBlock» considered critical supported in *«IT Block»* that support other «ISBlock» besides the critical ones (where an «IS Block» is considered critical if it supports exclusively critical «process») $\#\{ISBlock_{NC} \in ITBlock_C\}$ – is the number of «ISBlock» considered not critical supported in *«IT Block»* that support other «ISBlock» besides the non-critical ones (where an «IS Block» is considered not critical if it supports exclusively not critical «process») *#«IS Block»* – is the number of «IS Block»

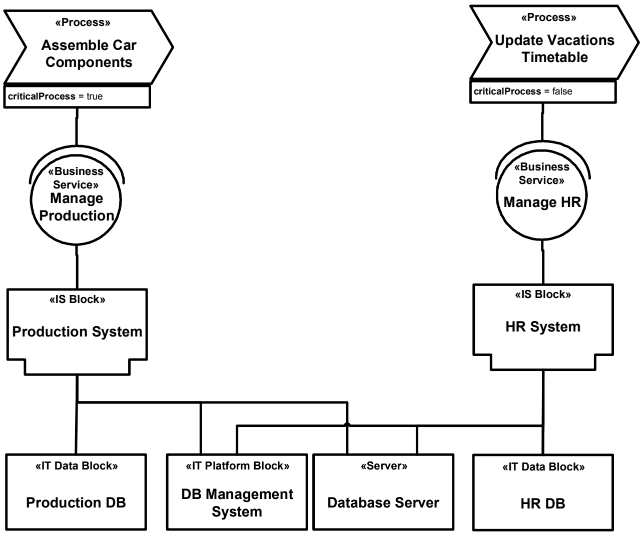
Formal Definition	`context EA_Specification::CSTMF(): Real`
	`  pre: self.TISB() > 0`
	`  post: result = 1 - self.TISBITBM() / self.TISB()`

Scale	[0; 1]
Arch. Levels	Application and Technology Architecture
ISA Primitives and attributes	Primitives: *«IS Block»*; *«IT Block; «Process» - attribute: criticalProcess*
ISA Qualities	The Application/Technology Alignment of an ISA tends to increase with this metric.
Support	In the same way that critical business processes should be supported by different applications than non-critical business processes Sousa, Pereira and Marques (2004), these applications should be implemented in technology components different than the ones that implement applications that support non-critical business processes.
Example	ISA A



$$CSTMF = 1 - \frac{0+0}{2} = 1$$

ISA B



$$CPSMF = 1 - \frac{1+1}{2} = 0$$