UNIVERSITÄT KOBLENZ · LANDAU

IWVI

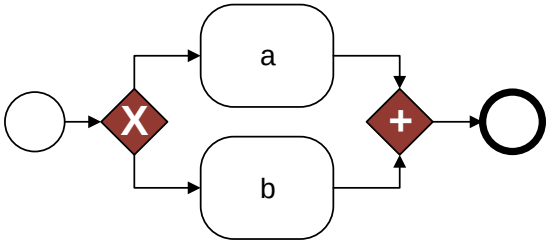
BUSINESS PROCESS MANAGEMENT

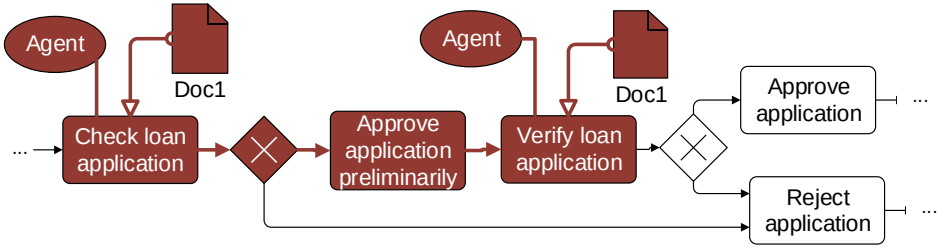
MODEL QUERY II:
THE GENERIC MODEL QUERY LANGUAGE (GRAPH MATCHING)

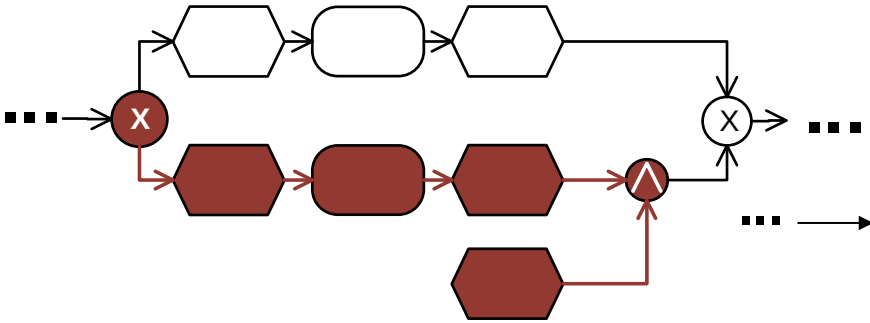
AGENDA

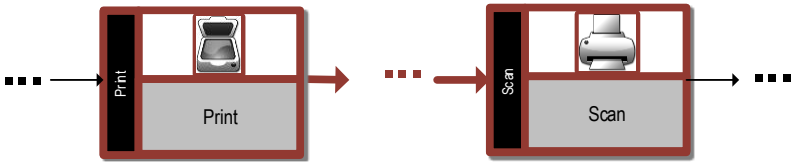
- Requirements of Model Query Languages
- The Generic Model Query Language (GMQL)
- Example Queries
- Live Demo

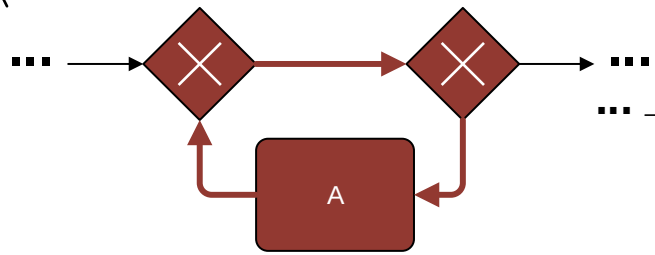
RECAP: TYPICAL STRUCTURES

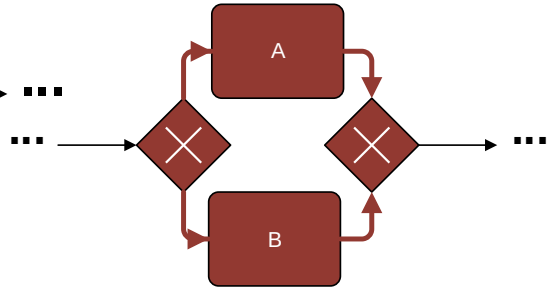


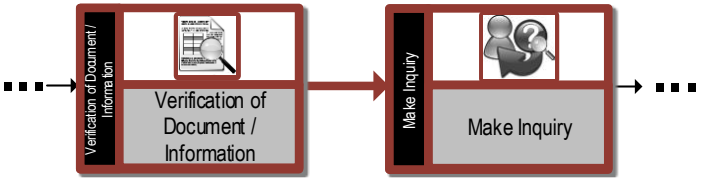


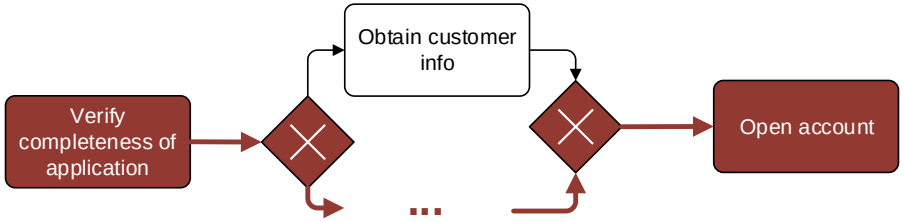






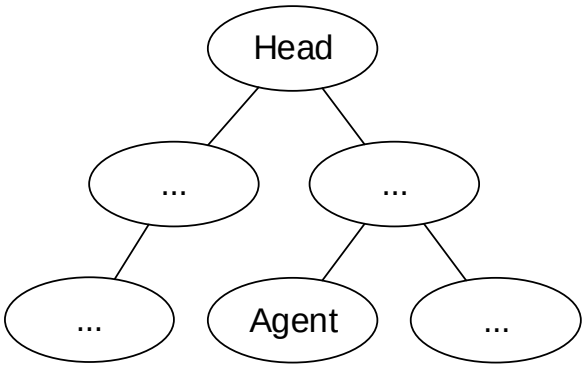




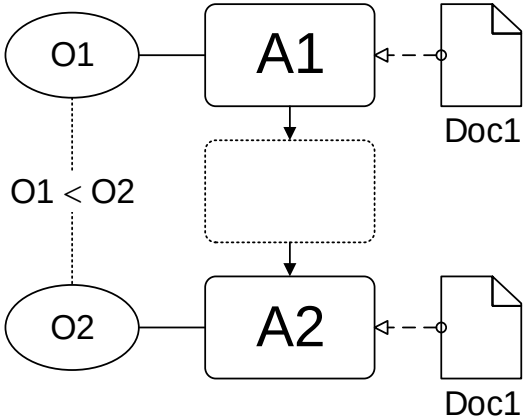


EXAMPLE: COMPLIANCE CHECKING

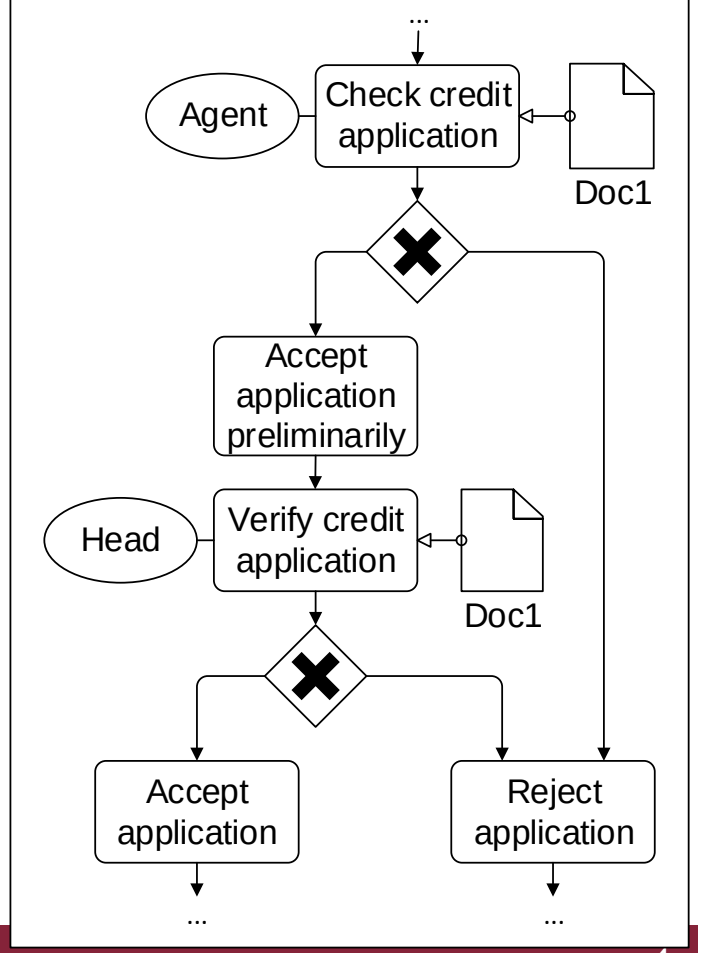
Organizational chart



Compliance rule *separation of duties*

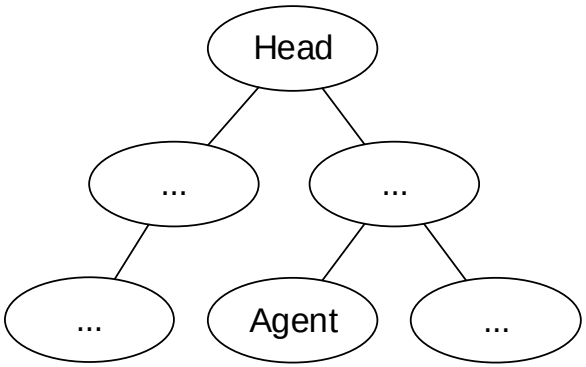


Process model

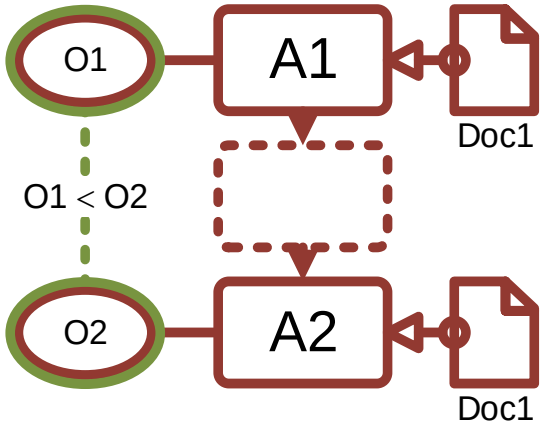


EXAMPLE: COMPLIANCE CHECKING

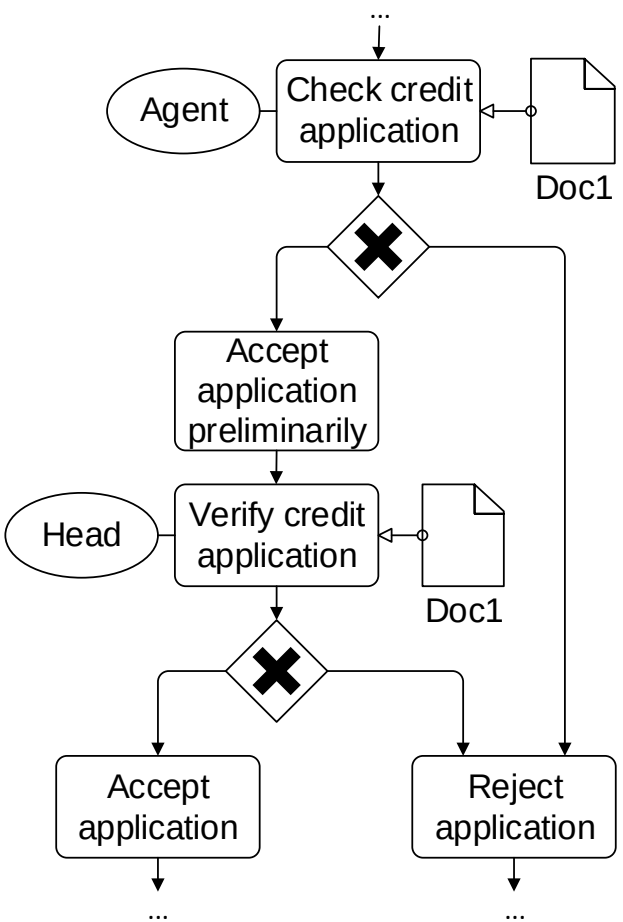
Organizational chart



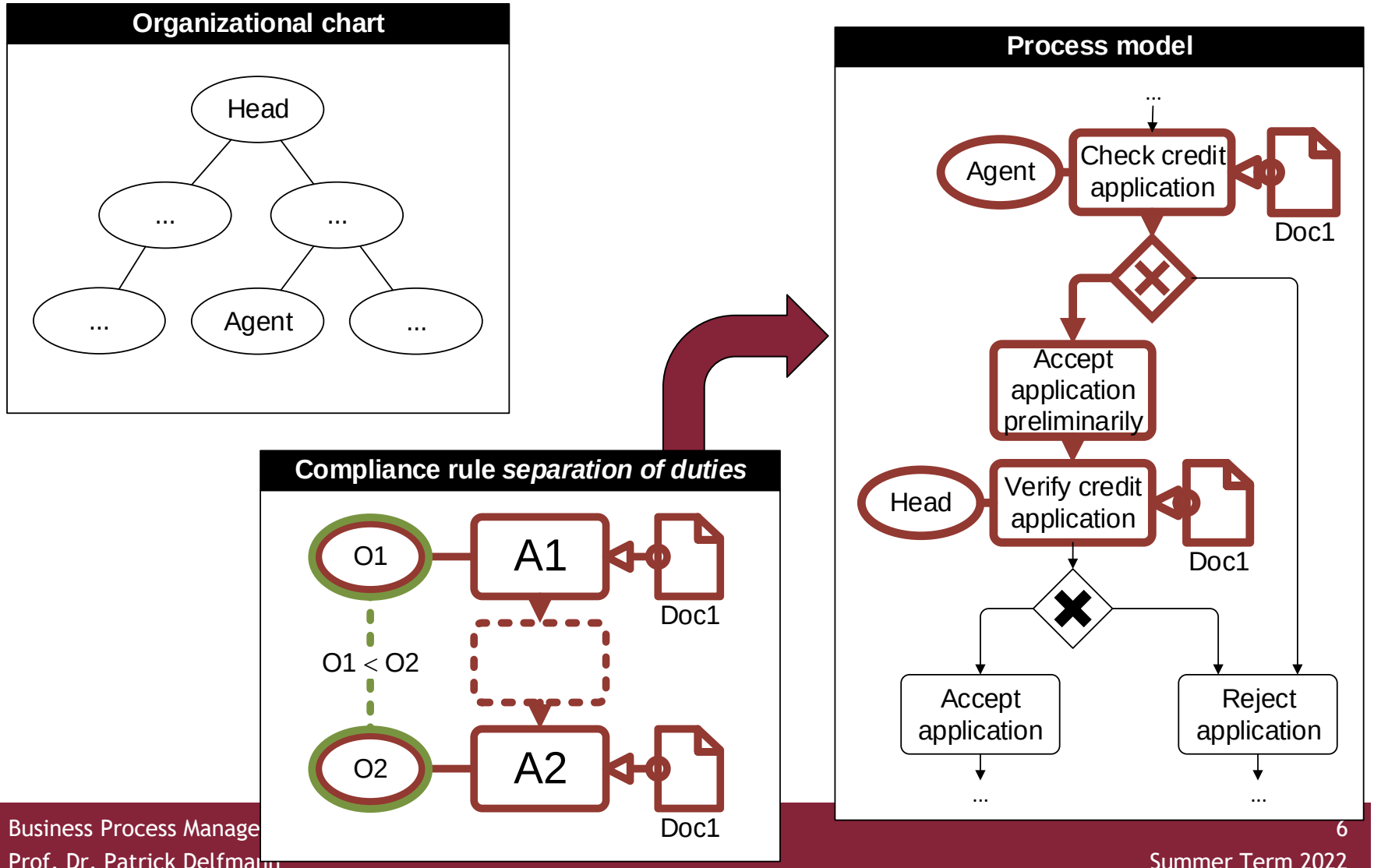
Compliance rule *separation of duties*



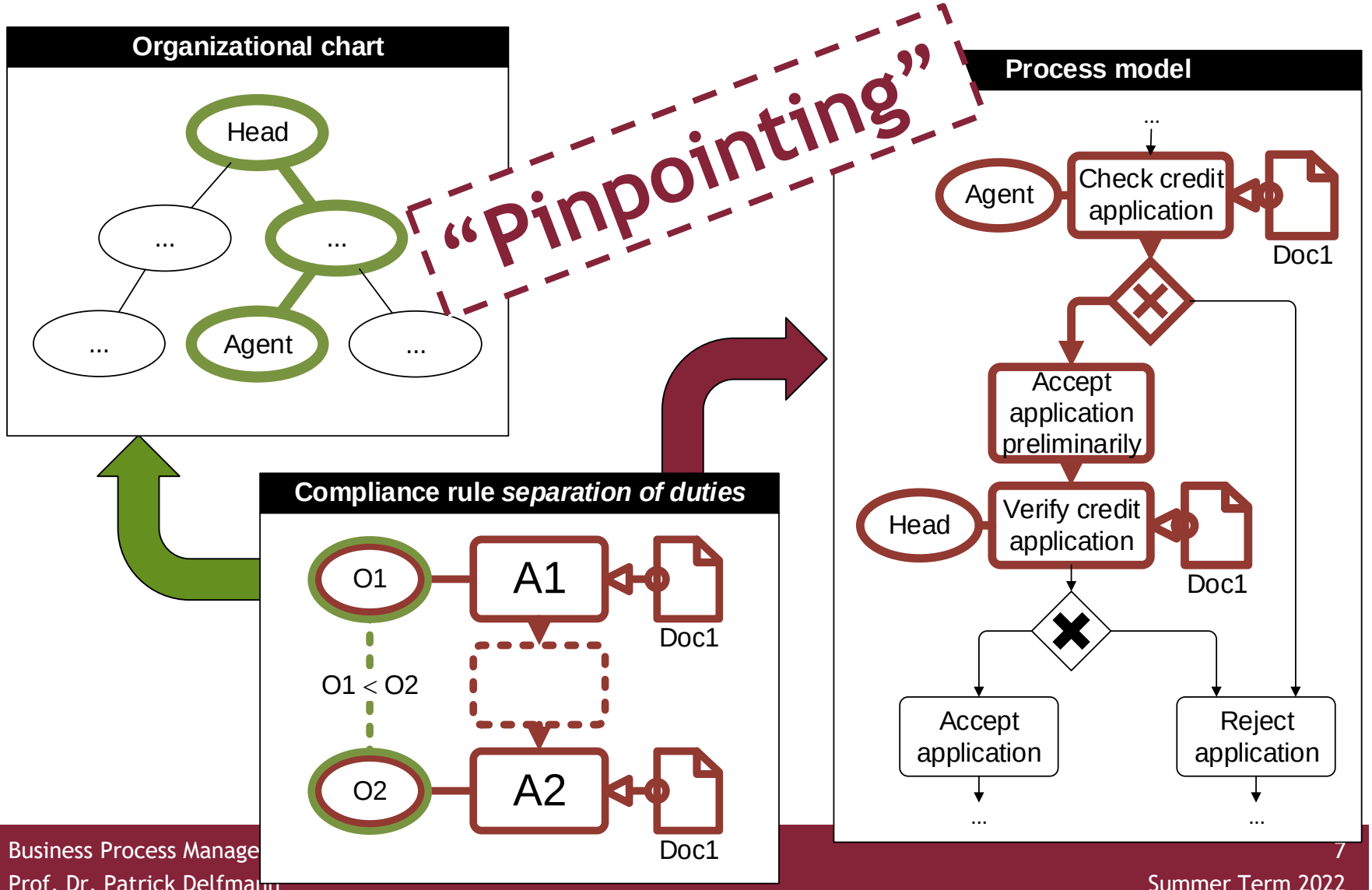
Process model



EXAMPLE: COMPLIANCE CHECKING



EXAMPLE: COMPLIANCE CHECKING



AGENDA

- Requirements of Model Query Languages
- The Generic Model Query Language (GMQL)
- Example Queries
- Live Demo

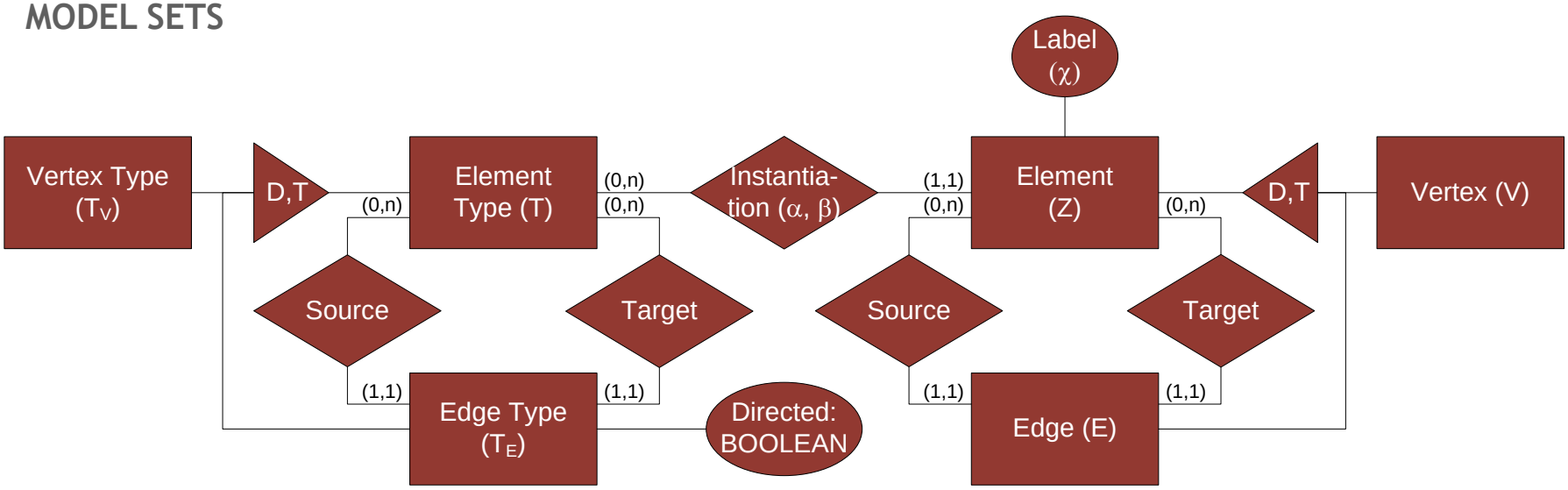
THE GENERIC MODEL QUERY LANGUAGE

BASIC IDEA

- Recognize any model as two basic sets
  - Set V of model vertices
  - Set E of model edges
- Provide set-altering functions and operators that perform operations on these basic sets
- Nest functions and operators to assemble a query

GMQL

MODEL SETS



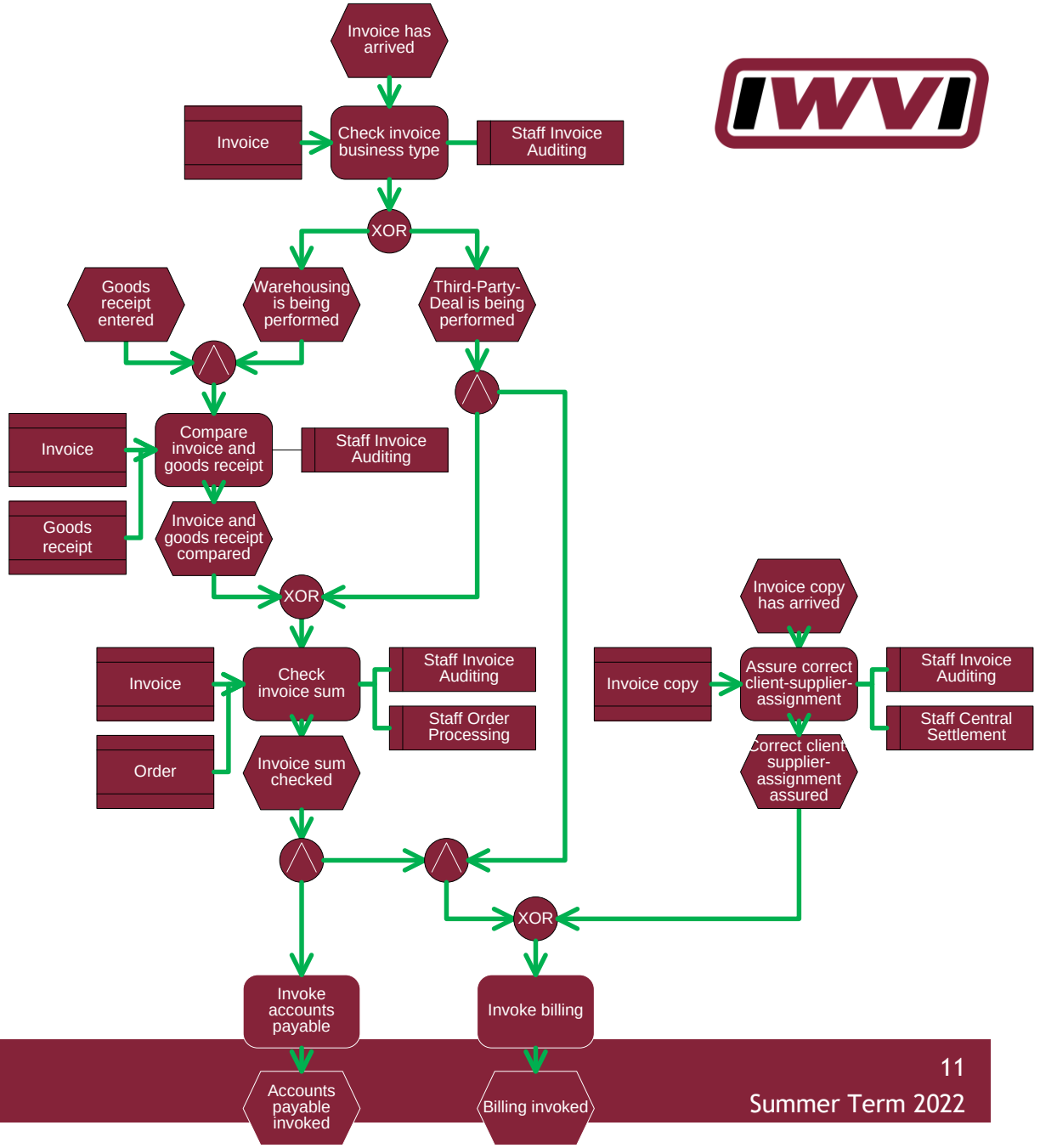
- Z: set of all elements available; $z \in Z$ is a particular element.
- V: set of all vertices available; $V \subseteq Z$; $v \in V$ is a particular vertex.
- E: set of all edges available; $E \subseteq Z$; $e \in E$ is a particular edge.
- T: set of all element types available; $t \in T$ is a particular element type.
- T_V: set of all vertex types available; $T_V \subseteq T$; $t_V \in T_V$ is a particular vertex type.
- T_E: set of all edge types available; $T_E \subseteq T$; $t_E \in T_E$ is a particular edge type.

GMQL

MODEL SETS EXAMPLES

- V
- E
- $Z=V \cup E$

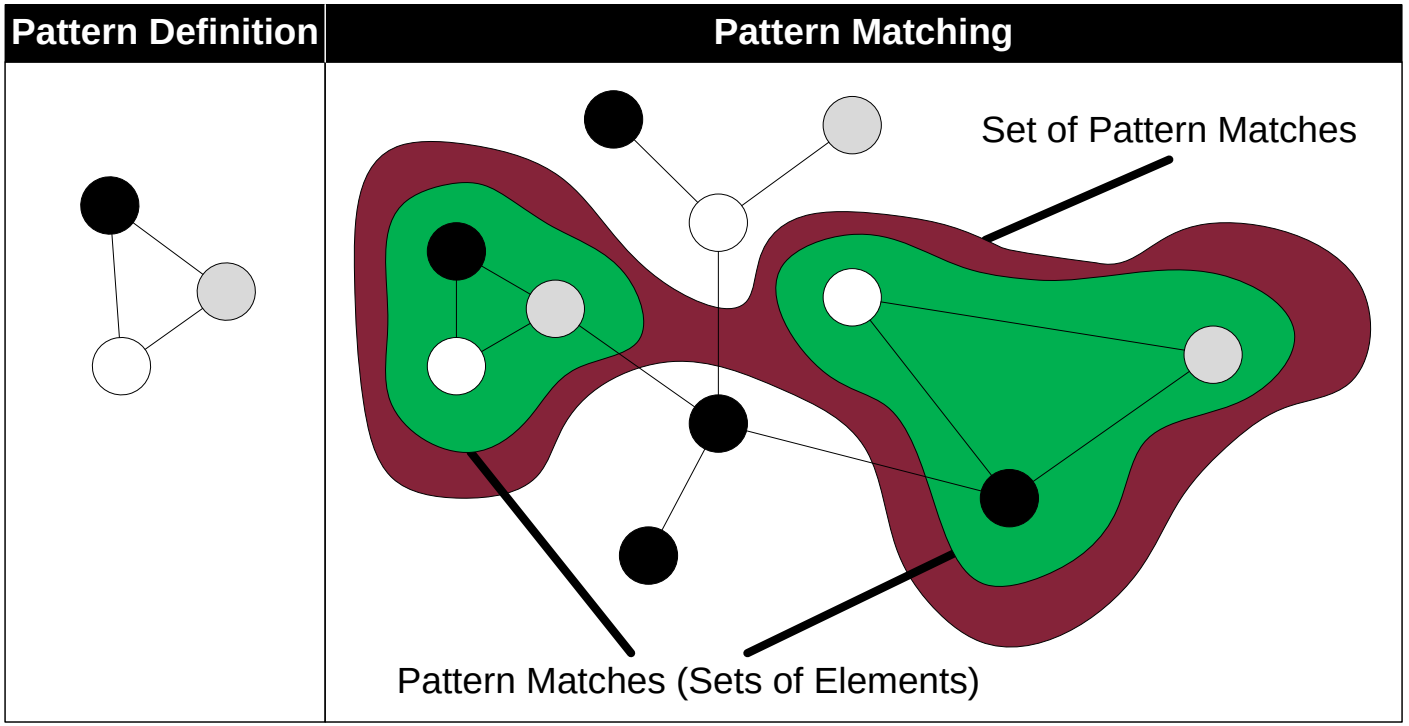
- T_V={event, function, xor, …}
- T_E={e_f, f_e, x_e, …}
- $T=T_V \cup T_E$



GMQL

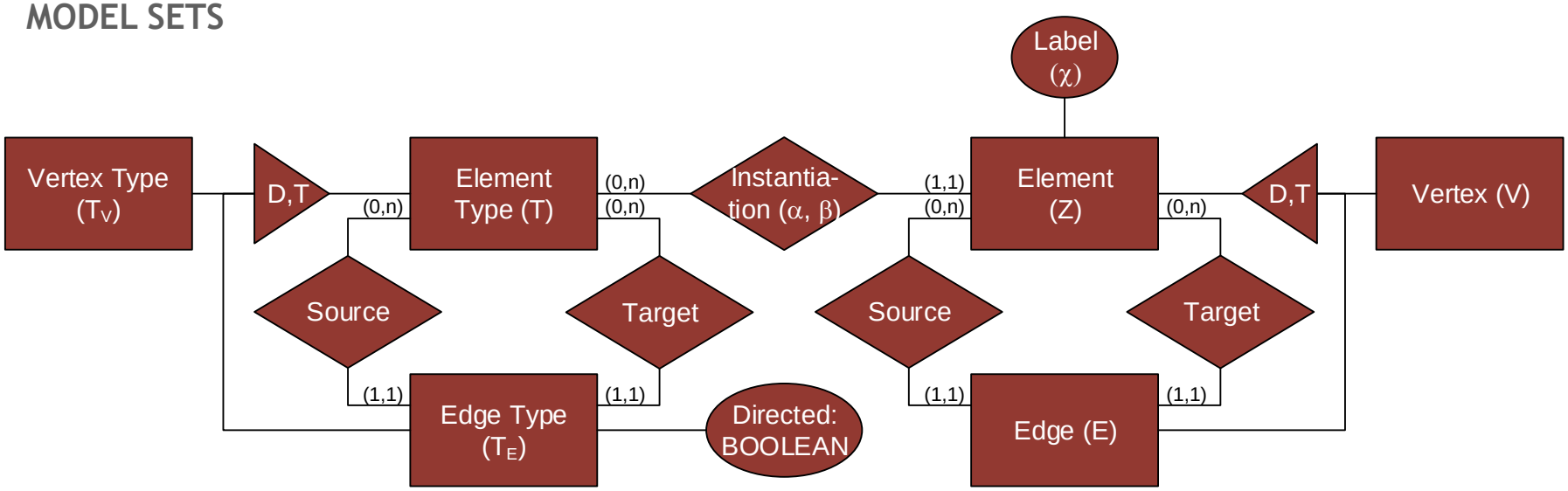
QUERY PROCESS: PATTERN MATCHING

- The pattern matching process returns a set of sets which is a subset of the Power Set of Z: $P(Z)$



GMQL

MODEL SETS



- I: set of all instantiations available; $I \subseteq T \times Z$;
  $(t,z) \in I$ is a particular instantiation.
- D: set of all edge destinations available; $D \subseteq Z \times E$;
  $(z,e) \in D$ is a particular destination.
- S: set of all edge sources available; $S \subseteq Z \times E$;
  $(z,e) \in S$ is a particular source.

GMQL

SOME MORE SETS

- X_k: any set of elements with $X_k \subseteq Z$
- x_l: a distinct element with $x_l \in Z$
- Y_V: any set of vertices with $y_V \in Y_V \subseteq V$.
- Y_E: any set of edges with $y_E \in Y_E \subseteq E$.
- *directed*(t_E): property *directed* of an edge type t_E.
- *datatype(v)*: property *data type* of a vertex *v*.
- *value(v)*: property *value* of a vertex *v*.
- n_x: a natural number $n_x \in N$

GMQL

SOME MORE SETS

- E_d: the set of all directed edges available; $E_d \subseteq E$, $((t_E, e_d) \in I \wedge directed(t_E) = TRUE \wedge t_E \in T_E) \ \forall e_d \in E_d$
- D_d: the set of all directed edge destinations available; $D_d \subseteq D$, $(e_d \in E_d) \ \forall (z, e_d) \in D$
- S_d: the set of all directed edge sources available; $S_d \subseteq S$, $(e_d \in E_d) \ \forall (z, e_d) \in S$
- D_u and S_u are undirected counterparts; $D_u = D \setminus D_d$ and $S_u = S \setminus S_d$

GMQL

1ST CLASS OF FUNCTIONS: ELEMENTS HAVING SPECIFIC ATTRIBUTES

- *ElementsOfType*$(X,t)=\{x \in X \mid (t,x) \in I\}$
  Returns a set containing all elements of *X* that belong to the given type

- *ElementsWithAttributeOfValue*$(X,t_V,u)=\{x \in X \mid t_V \in T_V, v \in V, e \in E, ((x,e) \in S \wedge (v,e) \in D) \vee ((v,e) \in S \wedge (x,e) \in D) \wedge (t_V,v) \in I \wedge value(v)=u\}$
  Returns a set containing all elements of *X* having an attribute of type t_V that carries the value u

- *ElementsWithAttributeOfDataType* $(X,u)=\{x \in X \mid v \in V, e \in E, ((x,e) \in S \wedge (v,e) \in D) \vee ((v,e) \in S \wedge (x,e) \in D) \wedge datatype(v)=u\}$
  Returns a set containing all elements of *X* having an attribute that is of datatype u

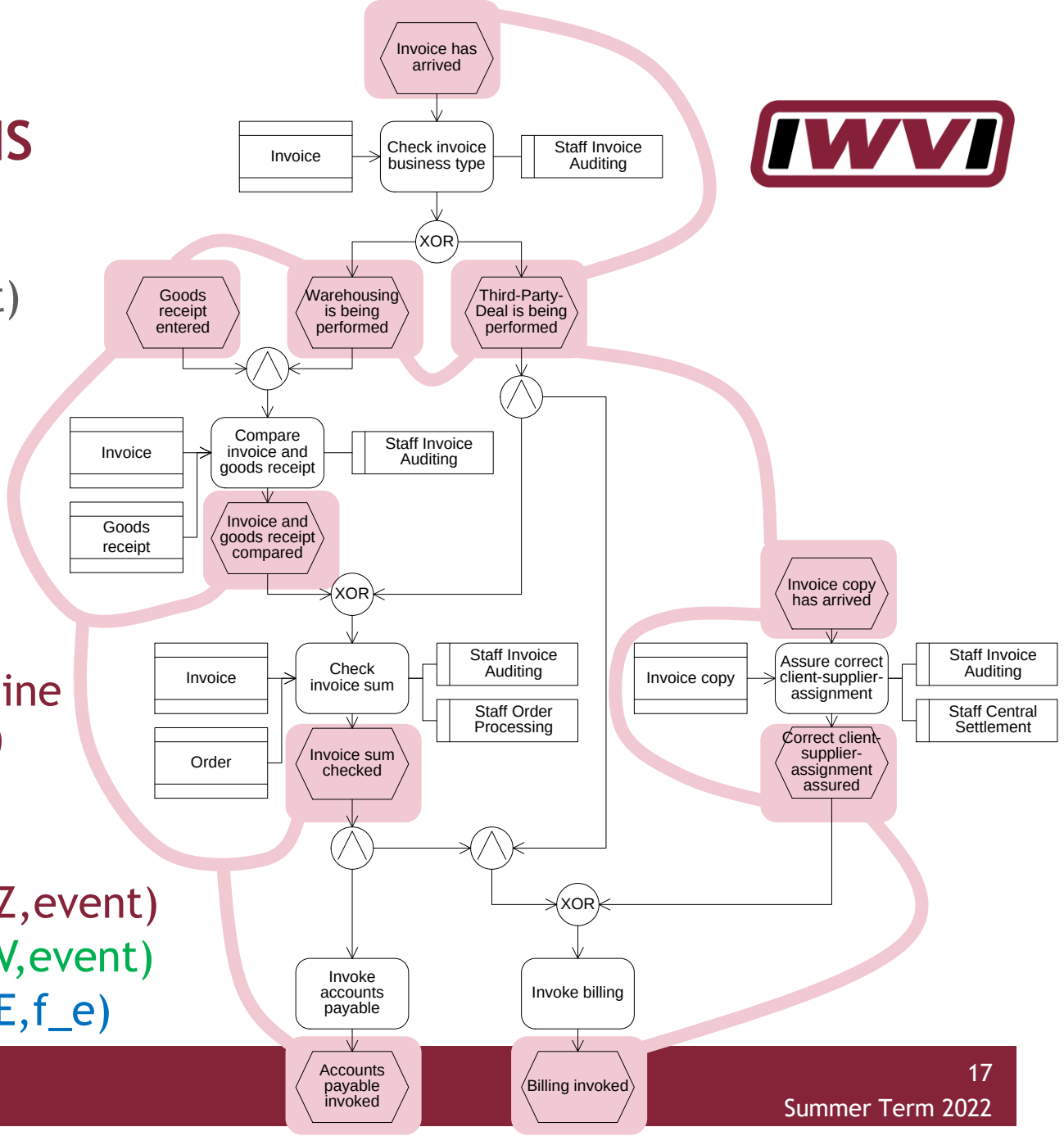
GMQL FUNCTIONS

EOT

- ElementsOfType(X,t)
  Returns a set containing all elements of *X* that belong to the given type

- Result: One simple set! (see single outline in the visualization)

- Examples
  - ElementsOfType(Z,event)
  - ElementsOfType(V,event)
  - ElementsOfType(E,f_e)



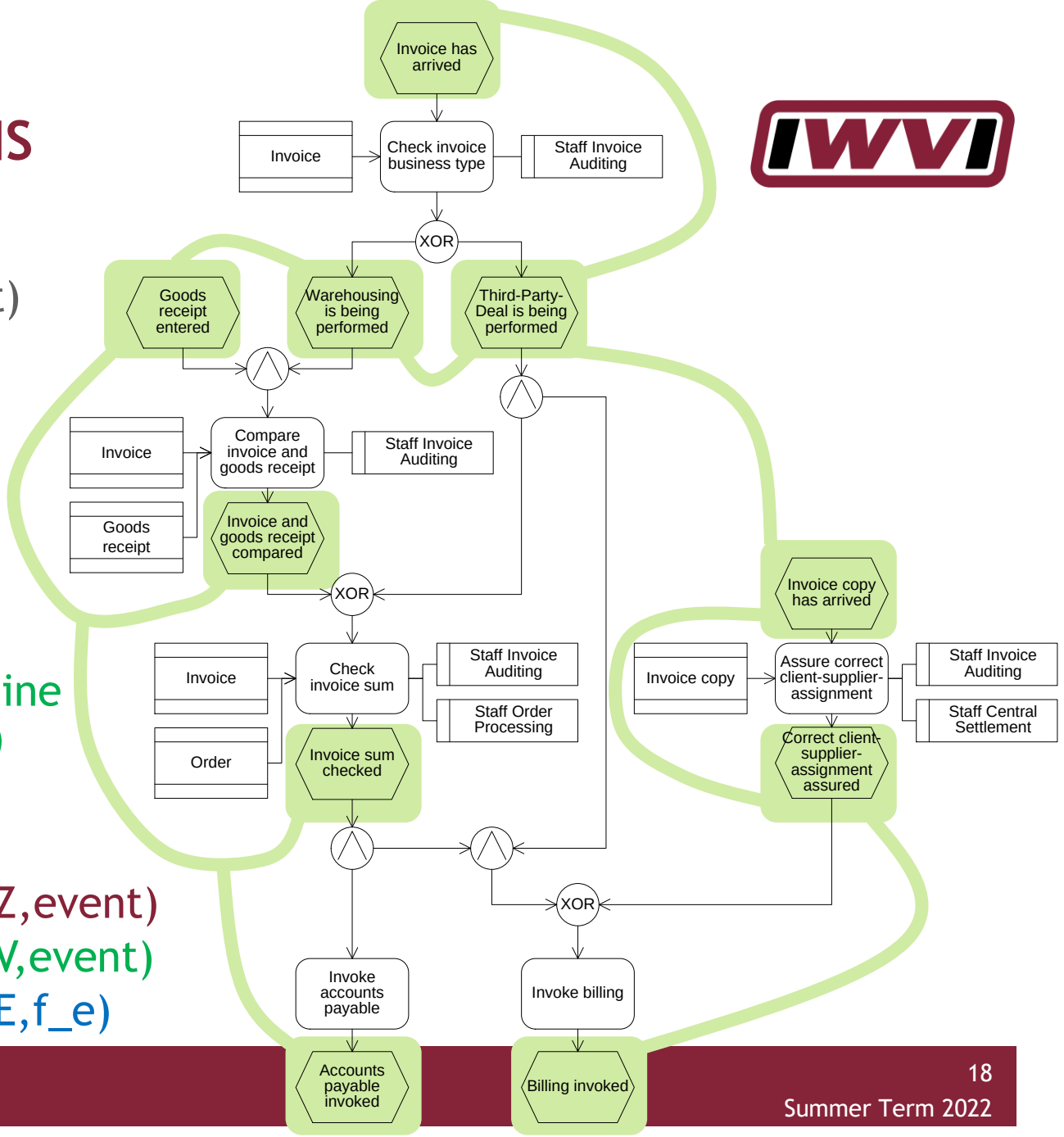
GMQL FUNCTIONS

EOT

- ElementsOfType(X,t)
  Returns a set containing all elements of *X* that belong to the given type

- Result: One simple set! (see single outline in the visualization)

- Examples
  - ElementsOfType(Z,event)
  - ElementsOfType(V,event)
  - ElementsOfType(E,f_e)



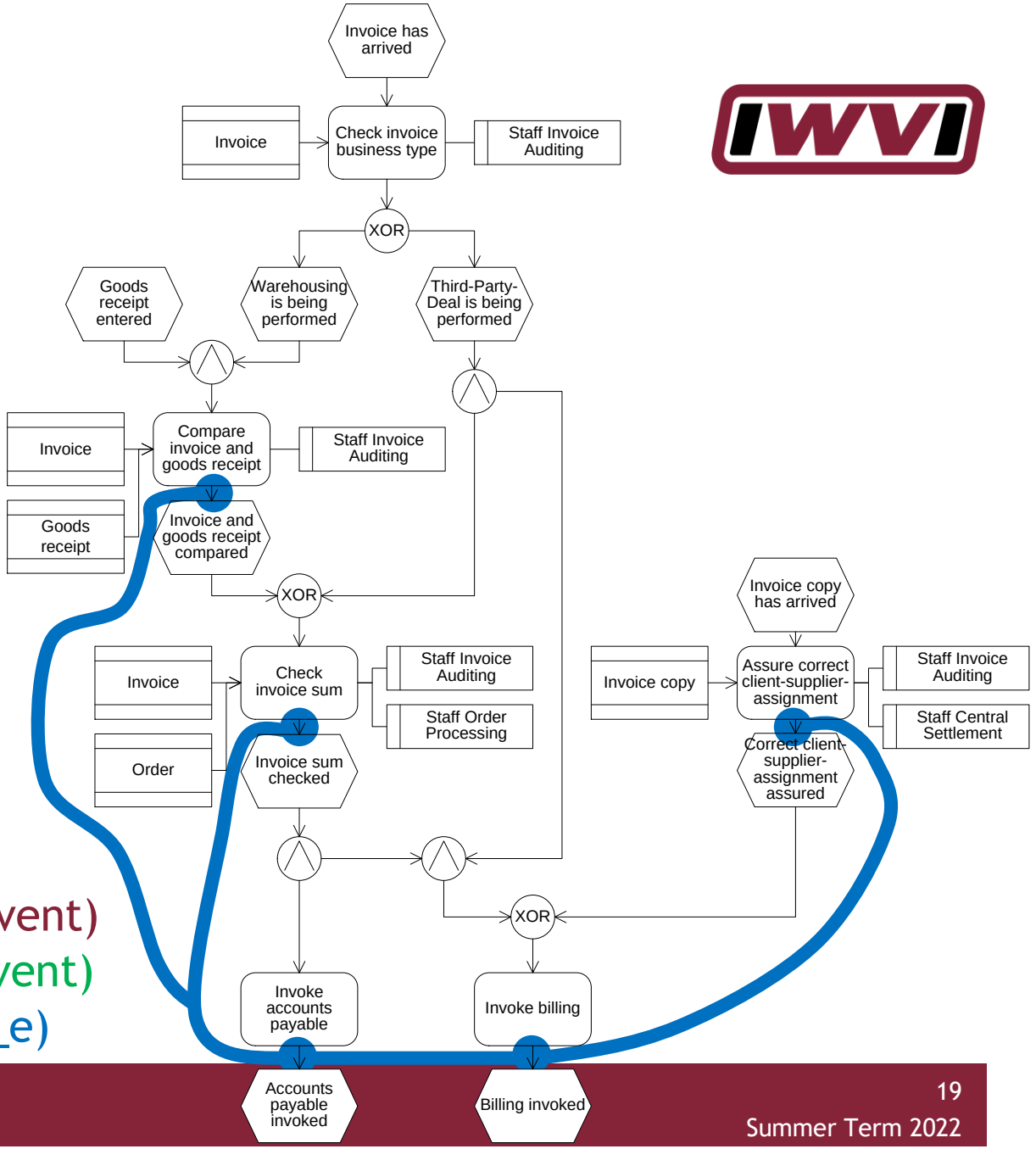
GMQL FUNCTIONS

EOT

- ElementsOfType(X,t)
  Returns a set containing all elements of *X* that belong to the given type

- Result: One simple set! (see single outline in the visualization)

- Examples
  - ElementsOfType(Z,event)
  - ElementsOfType(V,event)
  - ElementsOfType(E,f_e)

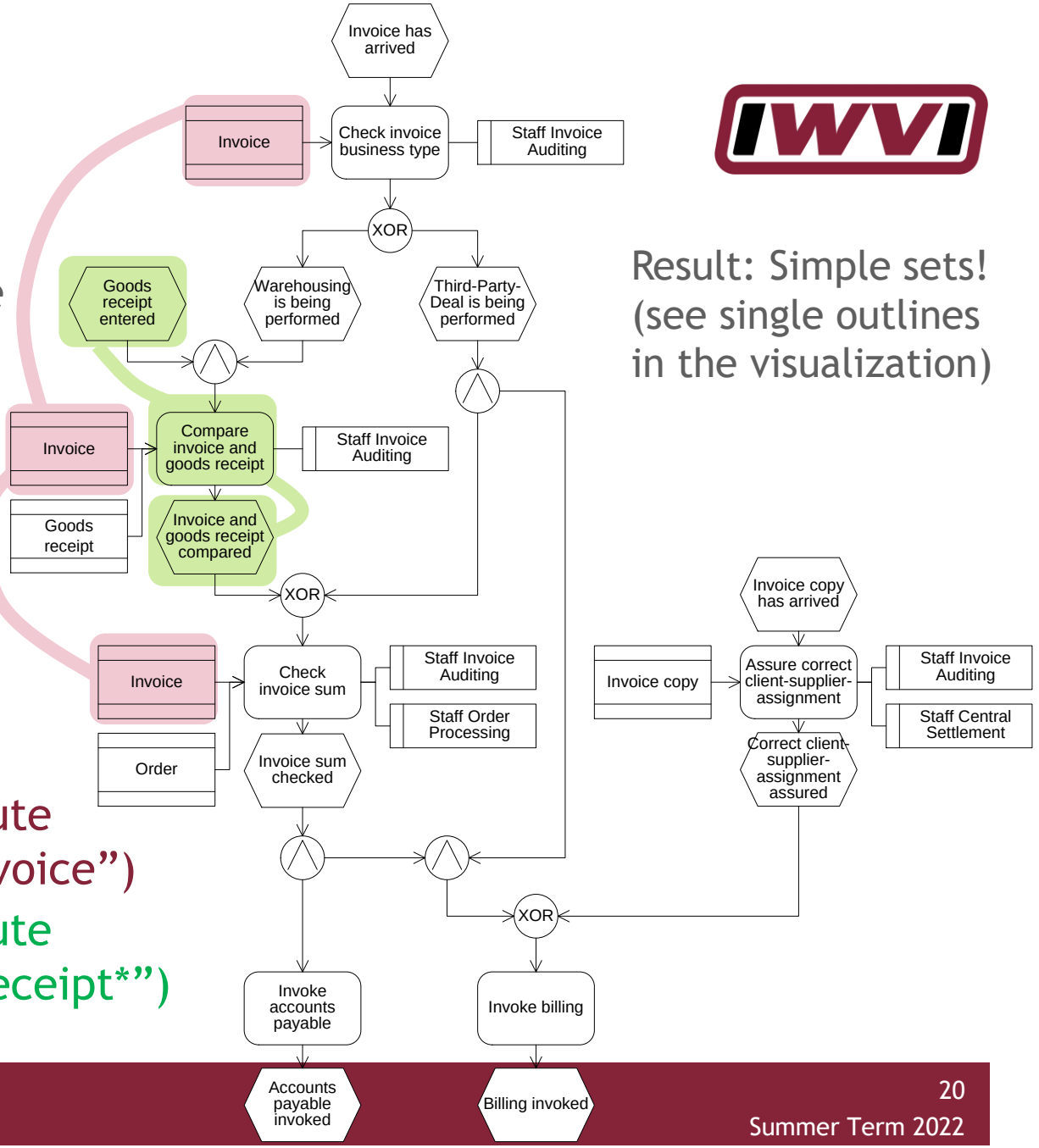


GMQL FUNCTIONS

EWAOV

- ElementsWithAttribute OfValue(X,t_V,u)
  Returns a set containing all elements of *X* having an attribute of type t_V that carries the value u

- Examples
  - ElementsWithAttribute OfValue (V,label,"Invoice")
  - ElementsWithAttribute OfValue (V,label,"*receipt*")



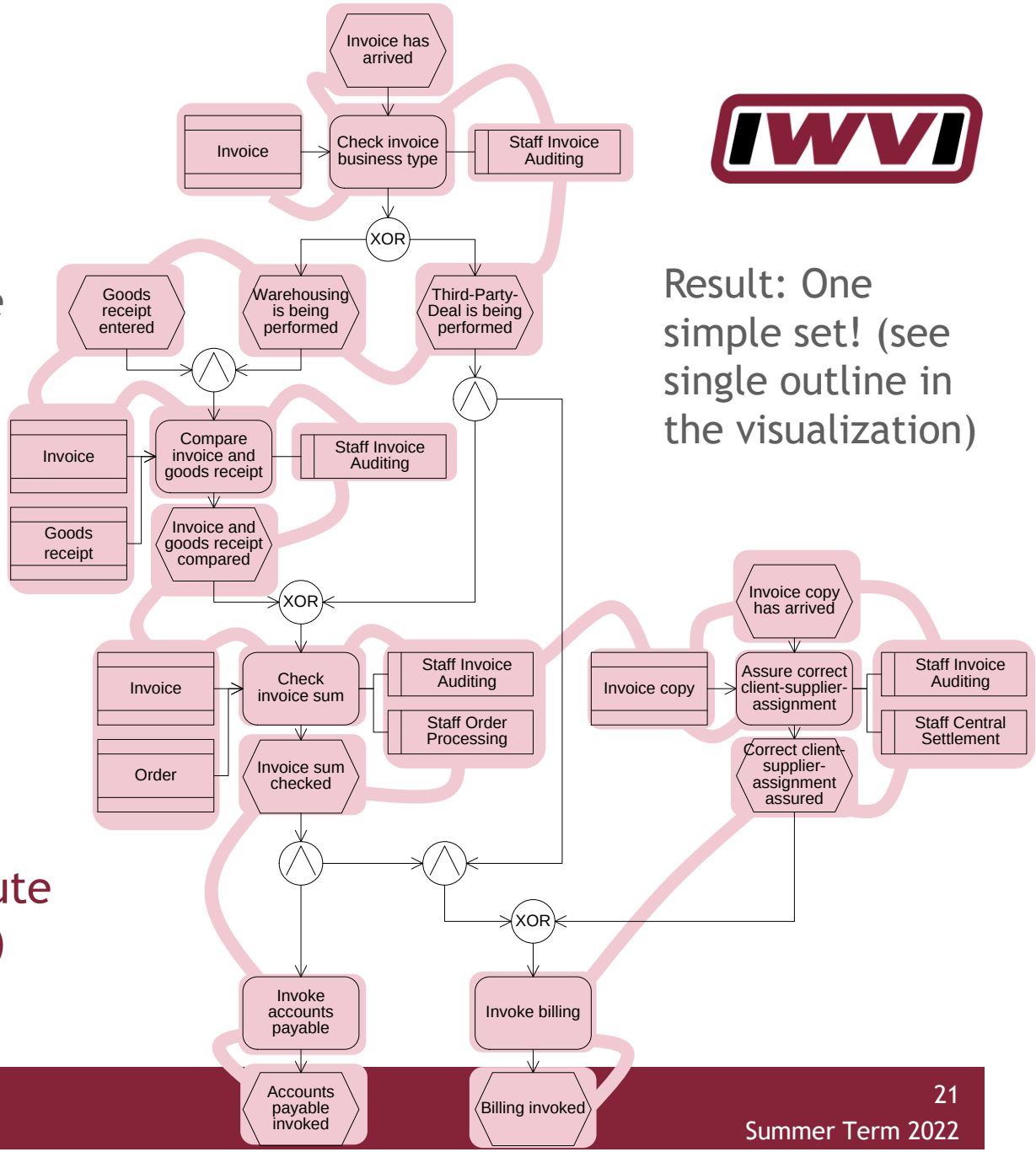
Result: Simple sets! (see single outlines in the visualization)

GMQL FUNCTIONS

EWAODT

- ElementsWithAttributeOfDataType(X,u)
  Returns a set containing all elements of *X* having an attribute that is of datatype u

- Example
  - ElementsWithAttributeOfDataType(V,string)

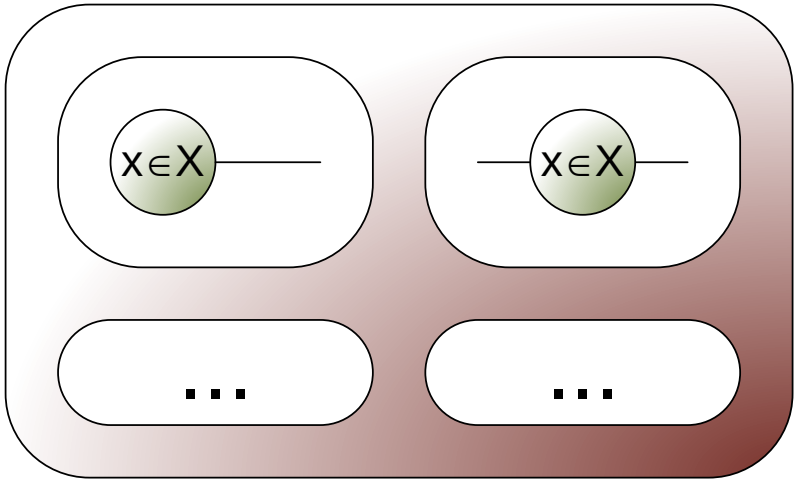


Result: One simple set! (see single outline in the visualization)

GMQL

2ND CLASS OF FUNCTIONS: ELEMENTS AND THEIR RELATIONSHIPS

- $ElementsWithRelations(X, Y_E) \subseteq P(Z)$
  Returns a set of sets. Each inner set contains an element of X and all its edges of Y_E.
- $ElWiRe(x_1, Y_E) = \{y_E \in Y_E \mid (x_1, y_E) \in D \vee (x_1, y_E) \in S\} \cup \{x_1\},\ x_1 \in X$
  $ElementsWithRelations(X, Y_E) = \{ElWiRe(x, Y_E)\},\ x \in X$

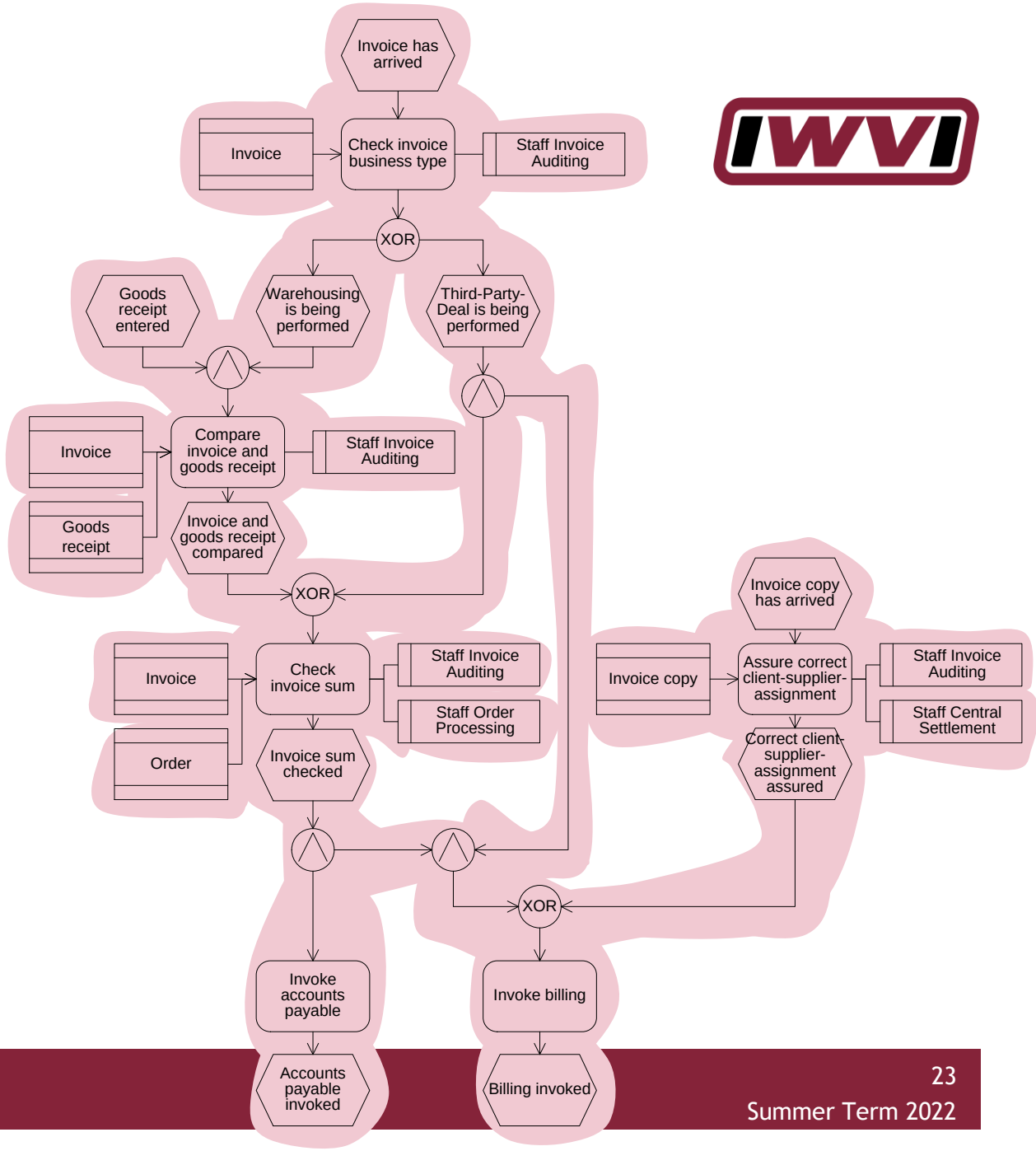


GMQL FUNCTIONS

EWR

- ElementsWith Relations(X, Y_E)
  Returns a set of sets. Each inner set contains an element of *X* and all its edges of Y_E.

- Examples
  - ElementsWith Relations(V,E)



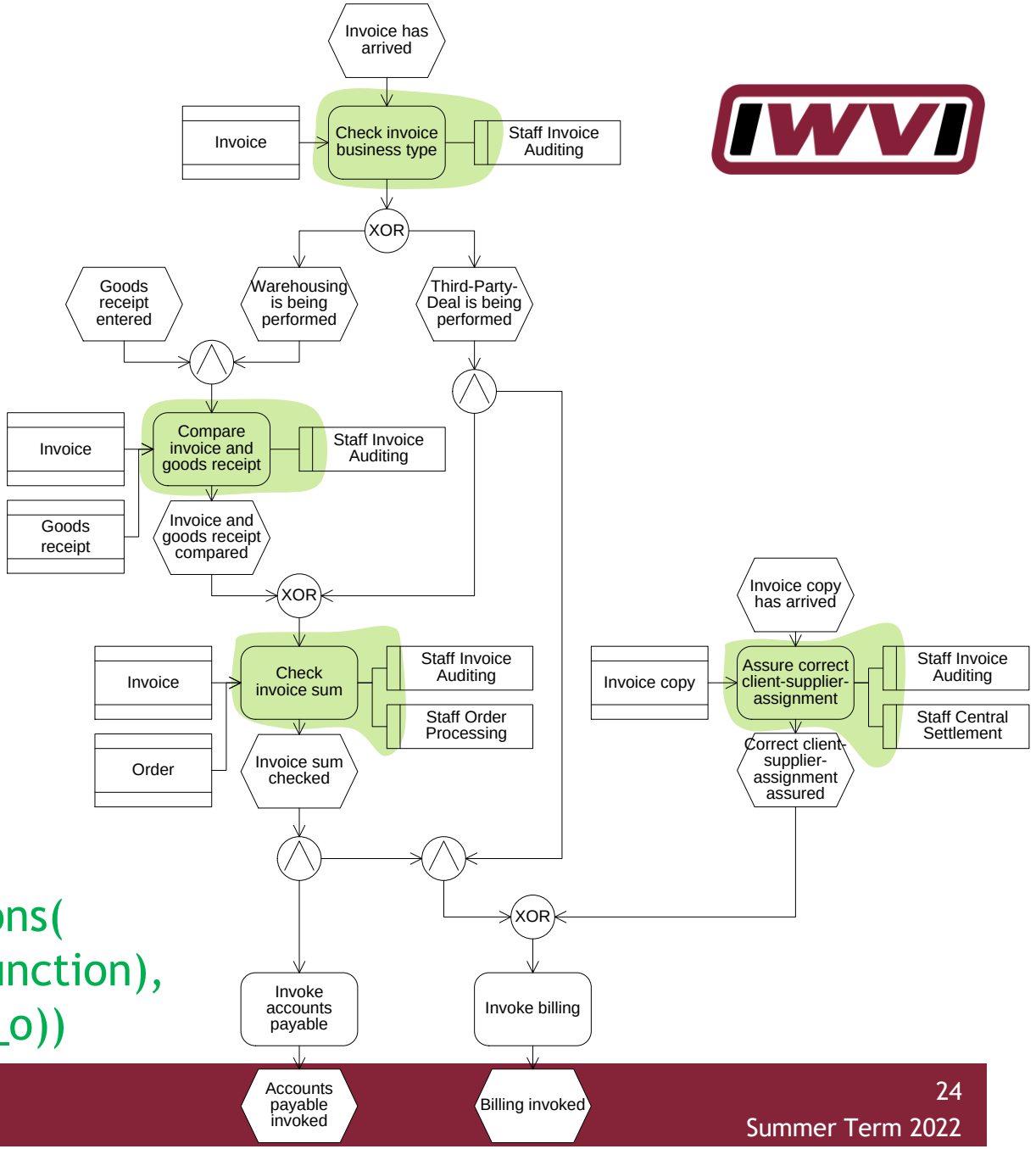
GMQL FUNCTIONS

EWR

- ElementsWith Relations(X, Y_E)
  Returns a set of sets. Each inner set contains an element of *X* and all its edges of Y_E.

- Examples

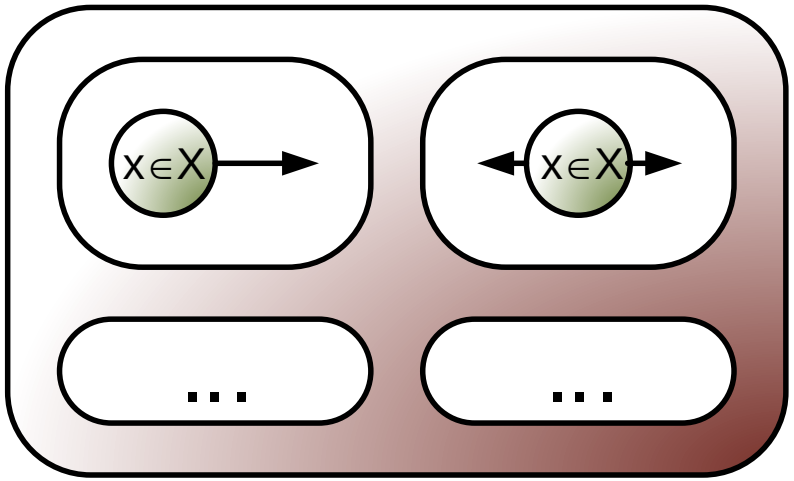
  - ElementsWithRelations(ElementsOfType(V,function), ElementsOfType(E,f_o))



GMQL

2ND CLASS OF FUNCTIONS: ELEMENTS AND THEIR RELATIONSHIPS

- $ElementsWithSuccRelations(X, Y_E) \subseteq P(Z)$
  Returns a set of sets. Each inner set contains an element of X an its outgoing edges of Y_E.
- $ElWiSuRe(x_1, Y_E) = \{y_E \in Y_E \mid (x_1, y_E) \in S_d\} \cup \{x_1\},\ x_1 \in X$
  $ElementsWithSuccRelations(X, Y_E) = \{ElWiSuRe(x, Y_E)\},\ x \in X$

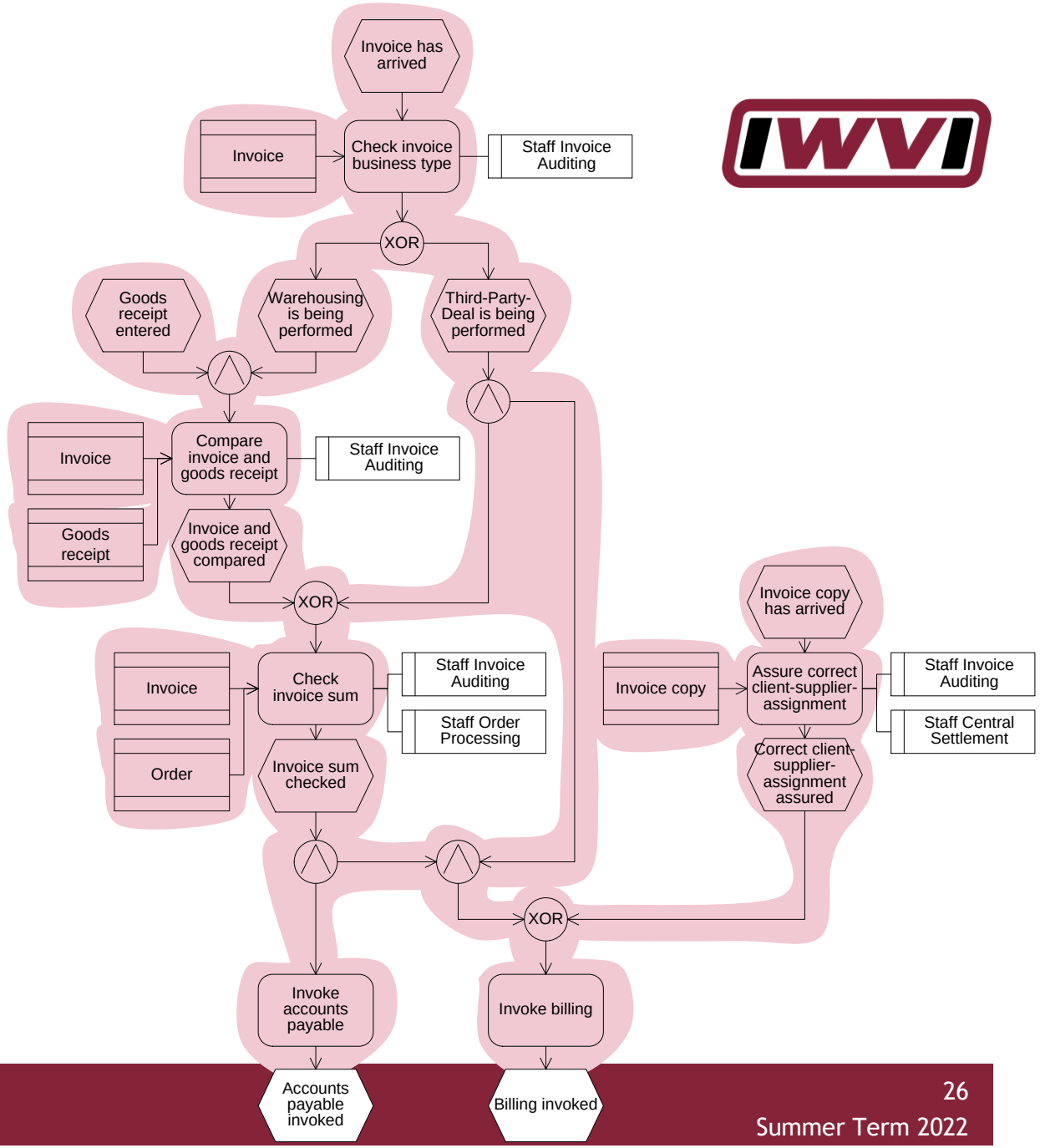


GMQL FUNCTIONS

EWSR

- *ElementsWith SuccRelations(X,Y_E)* Returns a set of sets. Each inner set contains an element of *X* an its outgoing edges of Y_E.

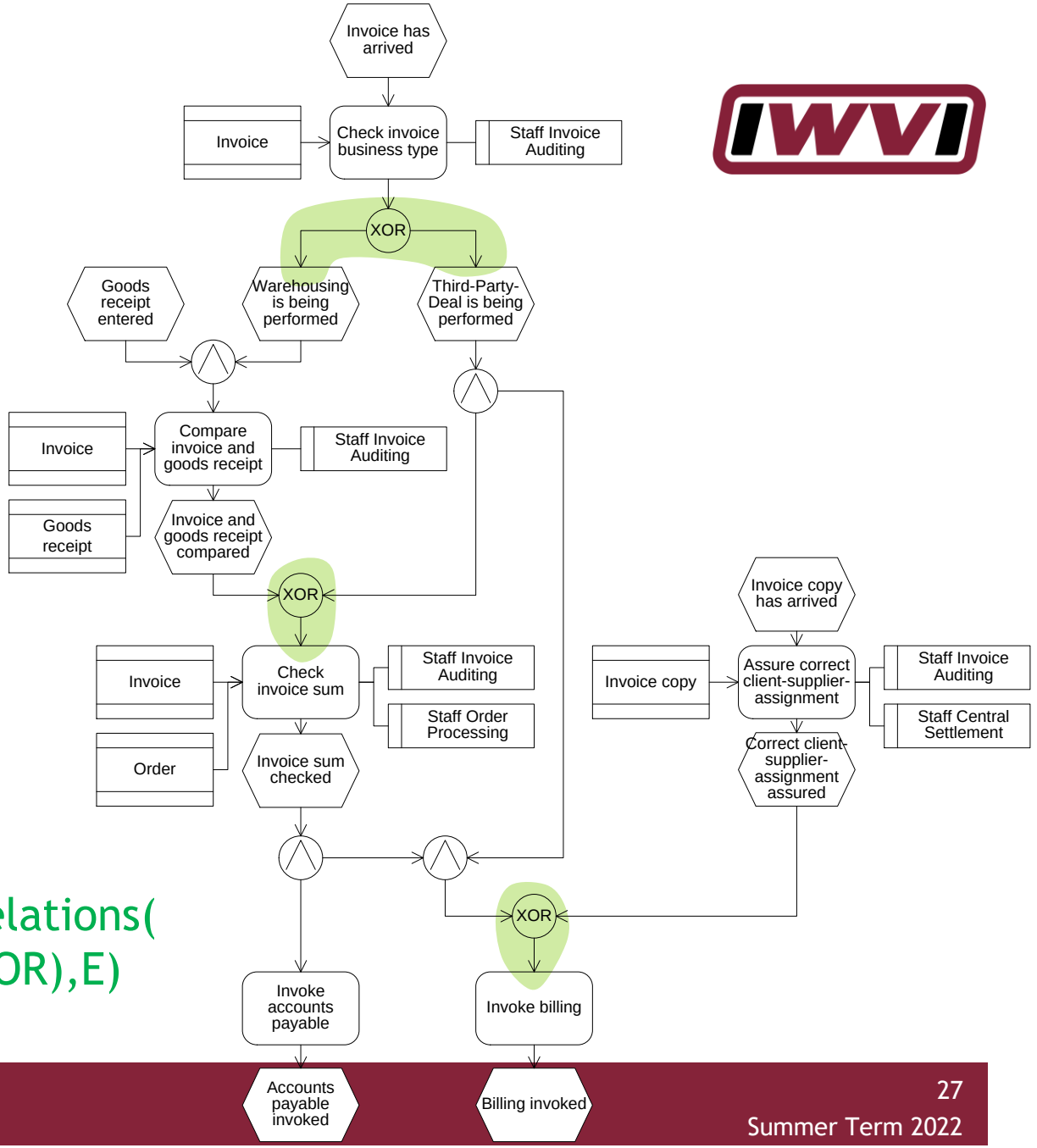
- Examples
  - ElementsWithSucc Relations(V,E)



GMQL FUNCTIONS

EWSR

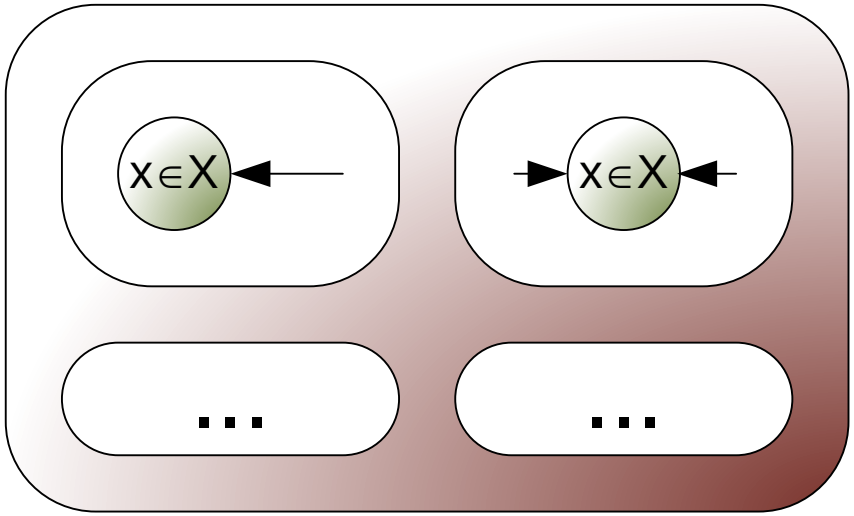
- *ElementsWith SuccRelations*(X, Y_E) Returns a set of sets. Each inner set contains an element of X an its outgoing edges of Y_E.
- Examples
  - ElementsWithSuccRelations(ElementsOfType(V,XOR),E)



GMQL

2ND CLASS OF FUNCTIONS: ELEMENTS AND THEIR RELATIONSHIPS

- $ElementsWithPredRelations(X, Y_E) \subseteq \mathcal{P}(Z)$
  Returns a set of sets. Each inner set contains an element of X and its incoming edges of Y_E.
- $ElWiPrRe(x_1, Y_E) = \{y_E \in Y_E \mid (x_1, y_E) \in D_d\} \cup \{x_1\},\ x_1 \in X$
  $ElementsWithPredRelations(X, Y_E) = \{ElWiPrRe(x, Y_E)\},\ x \in X$

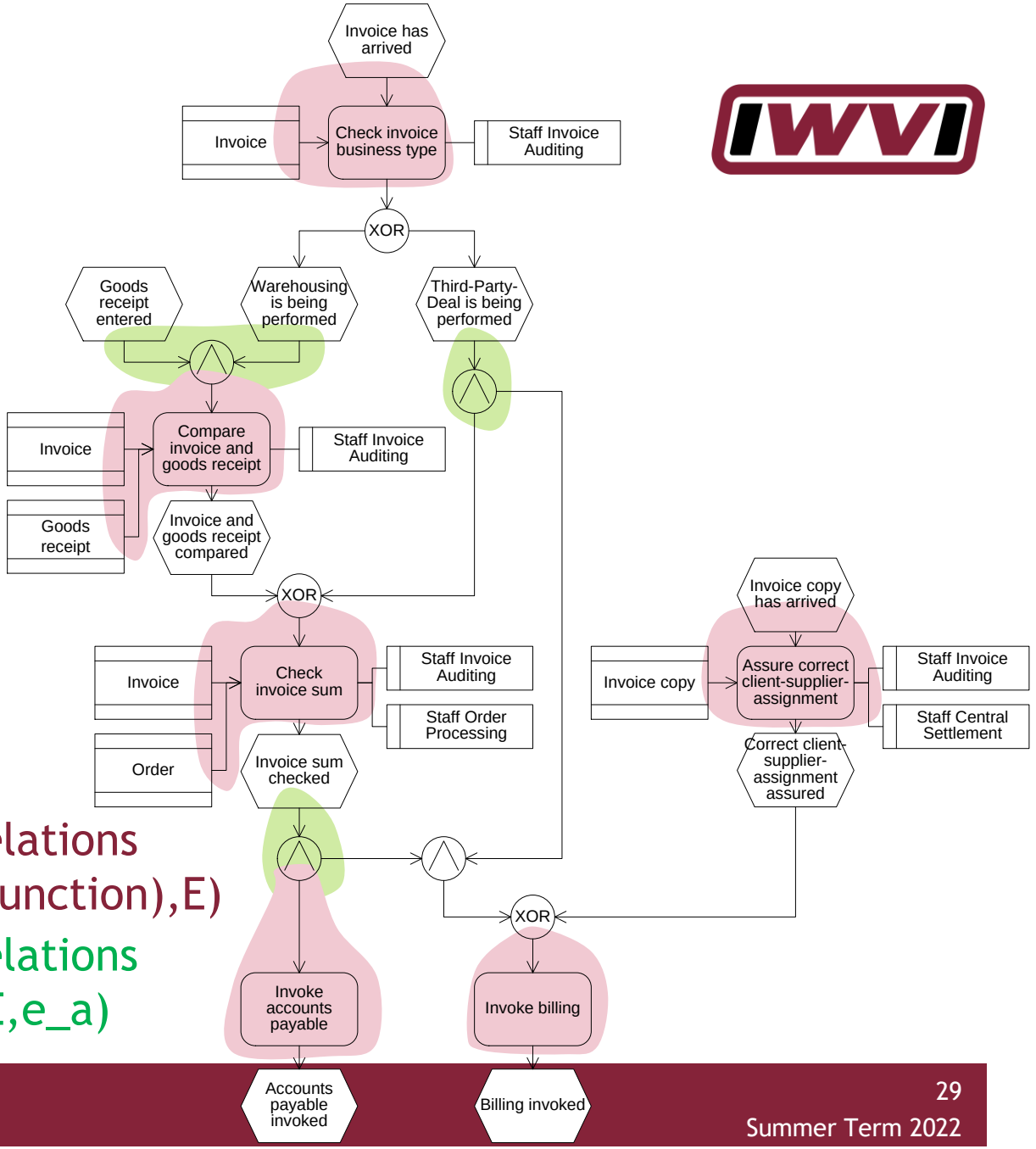


GMQL FUNCTIONS

EWPR

- ElementsWithPredRelations(X, Y_E)
  Returns a set of sets. Each inner set contains an element of X and its incoming edges of Y_E.

- Examples
  - ElementsWithPredRelations (ElementsOfType(V,function),E)
  - ElementsWithPredRelations (V,ElementsOfType(E,e_a)



GMQL

2ND CLASS OF FUNCTIONS: ELEMENTS AND THEIR RELATIONSHIPS

- *ElementsWithRelationsOfType*$(X, Y_E, t_E) \subseteq \mathcal{P}(Z)$
  Returns a set of sets. Each inner set contains an element of X and its edges of Y_E that are of type t_E.
  ElementsWithRelationsOfType(X, Y_E, t_E)=
  ElementsWithRelations$(X$,*ElementsOfType*$(Y_E, t_E))$

- *ElementsWithSuccRelationsOfType*$(X, Y_E, t_E) \subseteq \mathcal{P}(Z)$
  Returns a set of sets. Each inner set contains an element of X and its outgoing edges of Y_E that are of type t_E.
  ElementsWithSuccRelationsOfType(X, Y_E, t_E)=
  ElementsWithSuccRelations$(X$,*ElementsOfType*$(Y_E, t_E))$

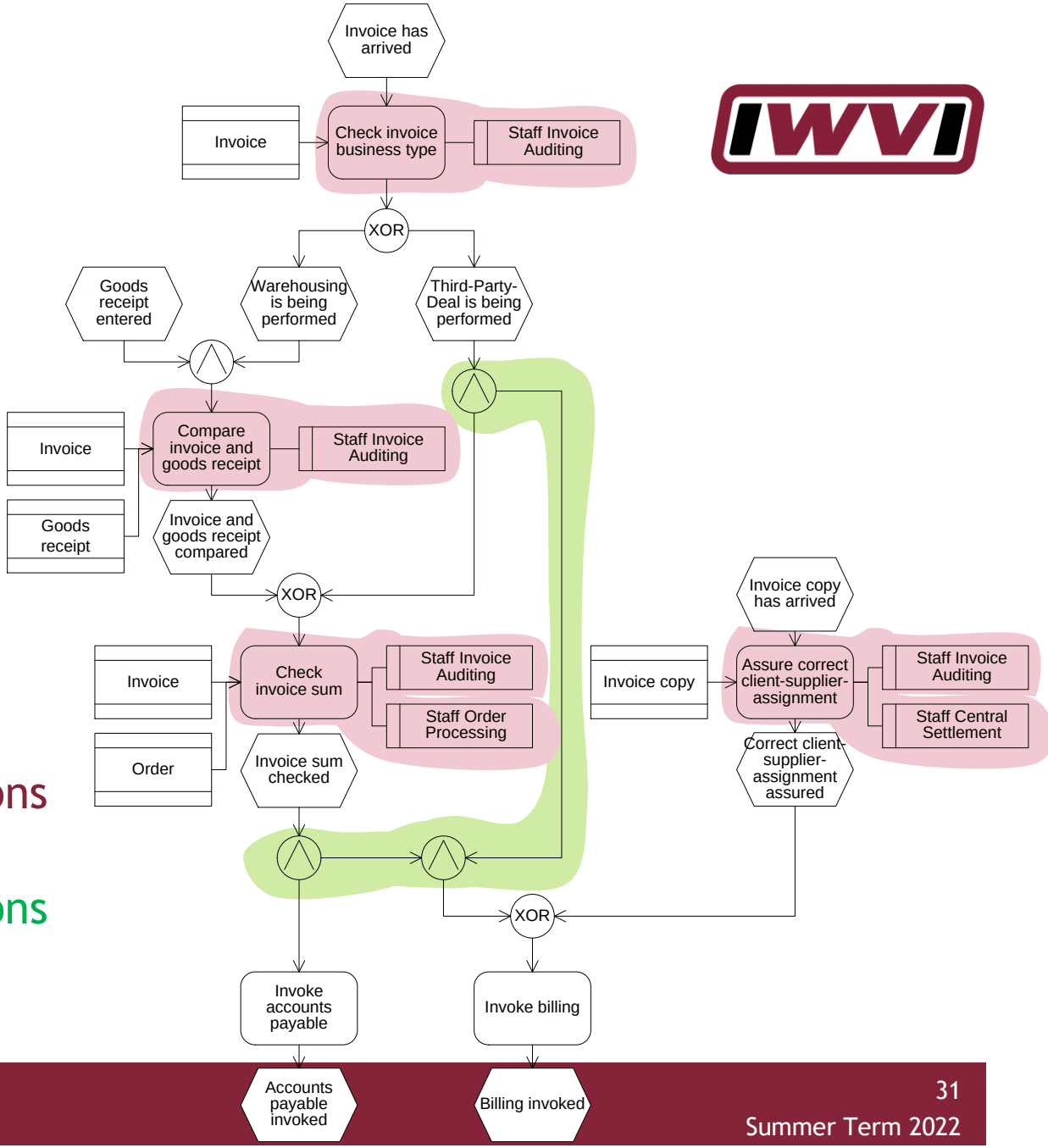
- *ElementsWithPredRelationsOfType*$(X, Y_E, t_E) \subseteq \mathcal{P}(Z)$
  Returns a set of sets. Each inner set contains an element of X and its ingoing edges of Y_E that are of type t_E.
  ElementsWithPredRelationsOfType(X, Y_E, t_E)=
  ElementsWithPredRelations$(X$,*ElementsOfType*$(Y_E, t_E))$

GMQL FUNCTIONS

EWROT

- ElementsWithRelations OfType(X, Y_E, t_E)
  Returns a set of sets. Each inner set contains an element of X and its edges of Y_E that are of type t_E.

- Examples
  - ElementsWithRelations OfType(V,E,f_o)
  - ElementsWithRelations OfType(V,E,a_a)

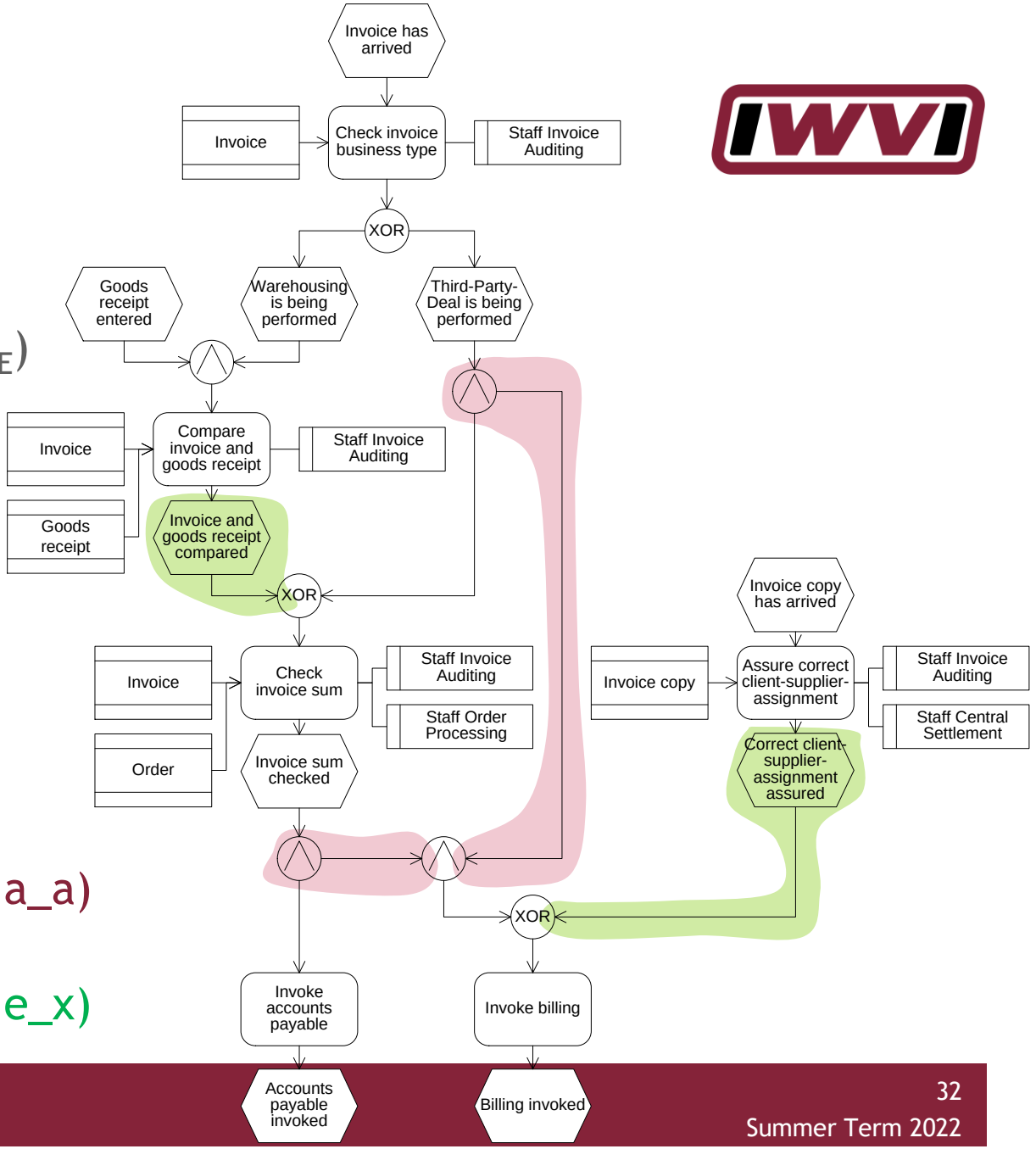


GMQL FUNCTIONS

EWSROT

- ElementsWithSucc RelationsOfType(X,Y_E,t_E)
  Returns a set of sets. Each inner set contains an element of *X* and its outgoing edges of Y_E that are of type t_E.

- Examples
  - ElementsWithSucc RelationsOfType(V,E,a_a)
  - ElementsWithSucc RelationsOfType(V,E,e_x)

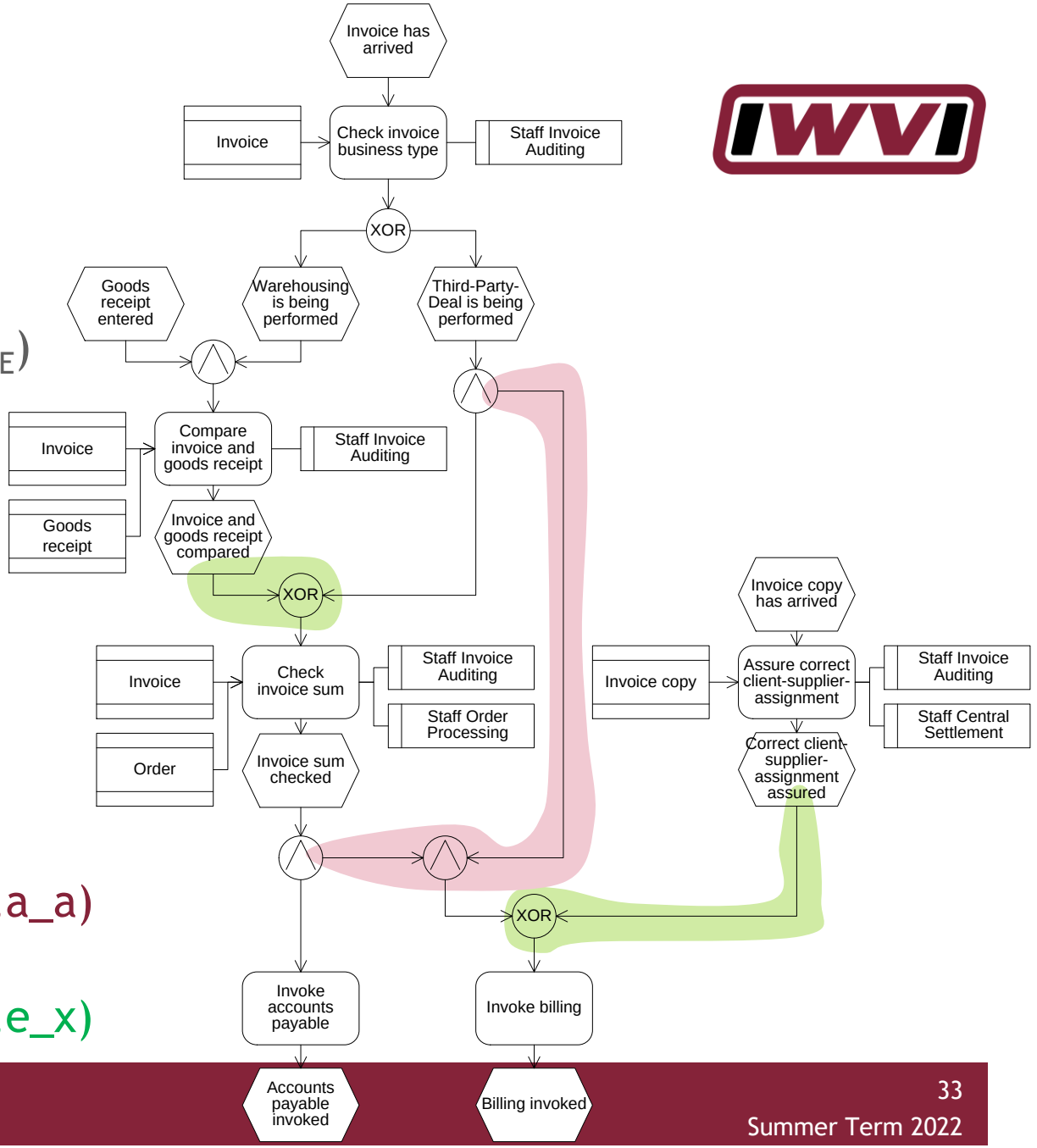


GMQL FUNCTIONS

EWPROT

- ElementsWithPred RelationsOfType(X,Y_E,t_E)
  Returns a set of sets. Each inner set contains an element of *X* and its ingoing edges of Y_E that are of type t_E.

- Examples
  - ElementsWithPred RelationsOfType(V,E,a_a)
  - ElementsWithPred RelationsOfType(V,E,e_x)



GMQL

2ND CLASS OF FUNCTIONS: ELEMENTS AND THEIR RELATIONSHIPS

- $ElementsWithNumberOfRelations(X,n_x) \subseteq P(Z)$
  Returns a set of sets. Each inner set contains an element of X and its n_x edges of E.
  $ElWiNuRe(x)=\{e \in E \mid (x,e) \in D \vee (x,e) \in S\} \cup \{x\}$
  $ElementsWithNumberOfRelations(X,n_x)=\{ElWiNuRe(x) \mid \mid ElWiNuRe\ (x)\mid =n_x+1\}$

- $ElementsWithNumberOfSuccRelations(X,n_x) \subseteq P(Z)$
  Returns a set of sets. Each inner set contains an element of X and its n_x outgoing edges of E.
  $ElWiNuSuRe(x)=\{e \in E_d \mid (x,e) \in S_d\} \cup \{x\}$
  $ElementsWithNumberOfSuccRelations(X,n_x)=\{ElWiNuSuRe(x) \mid \mid ElWiNuSuRe(x)\mid =n_x+1\}$

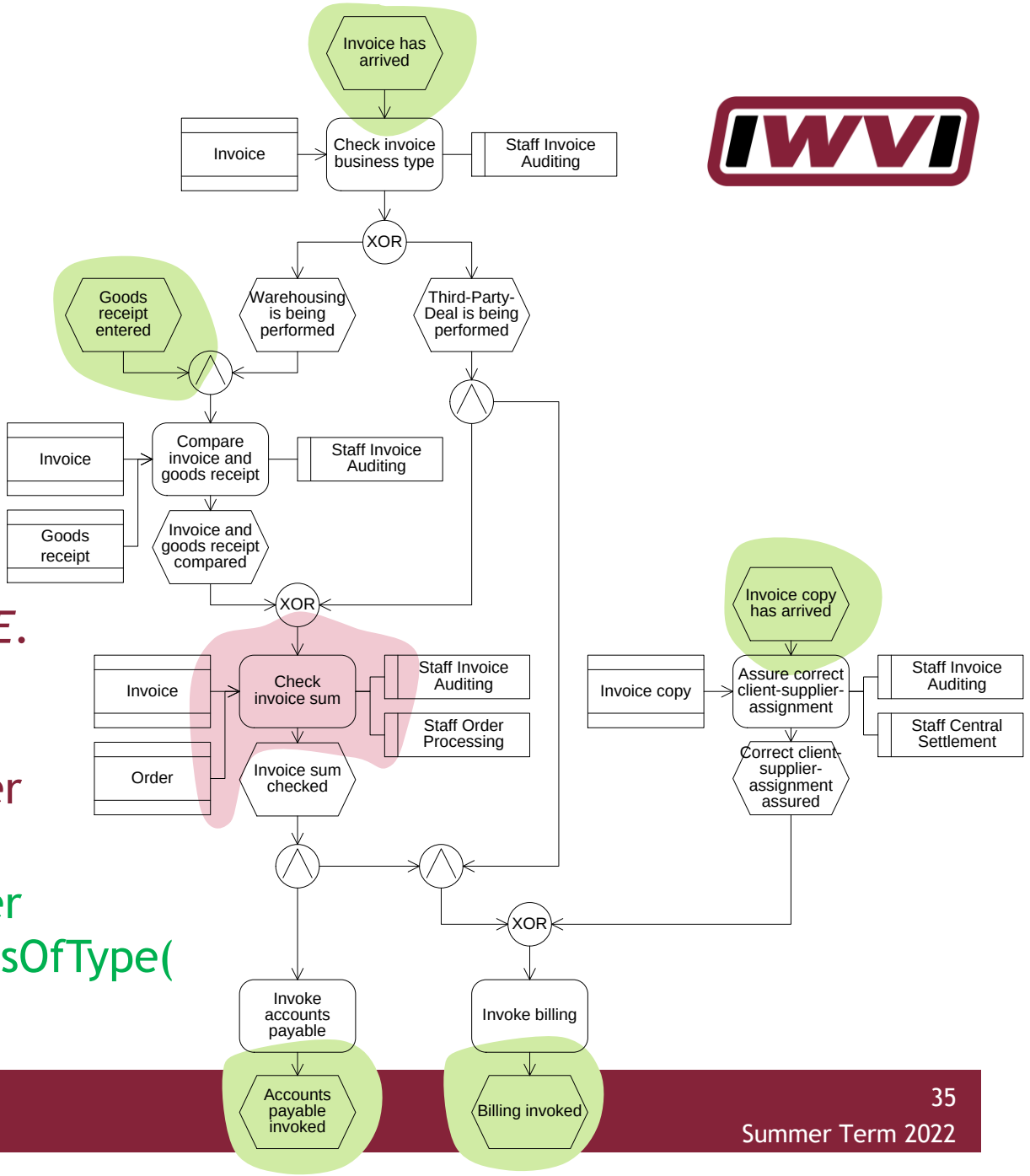
- $ElementsWithNumberOfPredRelations(X,n_x) \subseteq P(Z)$
  Returns a set of sets. Each inner set contains an element of X and its n_x incoming edges of E.
  $ElWiNuPrRe(x)=\{e \in E_d \mid (x,e) \in D_d\} \cup \{x\}$
  $ElementsWithNumberOfSuccRelations(X,n_x)=\{ElWiNuPrRe(x) \mid \mid ElWiNuPrRe(x)\mid =n_x+1\}$

GMQL FUNCTIONS

EWNOR

- ElementsWithNumberOfRelations(X,n_x)
  Returns a set of sets. Each inner set contains an element of *X* and its edges, if it is related to n_x edges of *E*.

- Examples
  - ElementsWithNumberOfRelations(V,6)
  - ElementsWithNumberOfRelations(ElementsOfType(V,event),1)

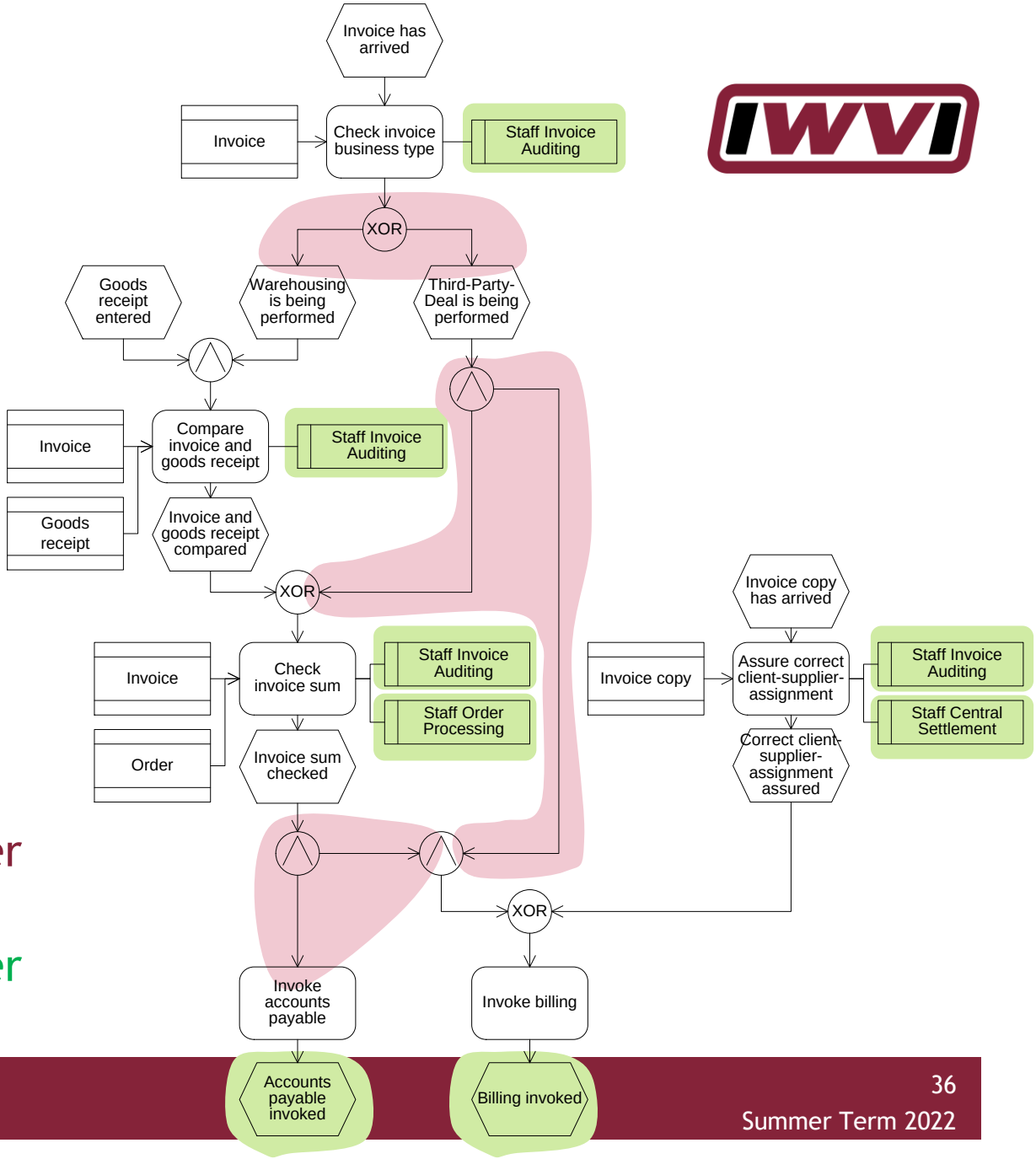


GMQL FUNCTIONS

EWNOSR

- ElementsWithNumberOfSuccRelations(X,n_x)
  Returns a set of sets. Each inner set contains an element of *X* and its edges, if it is related to n_x outgoing edges of *E*.

- Examples
  - ElementsWithNumberOfSuccRelations(V,2)
  - ElementsWithNumberOfSuccRelations(V,0)

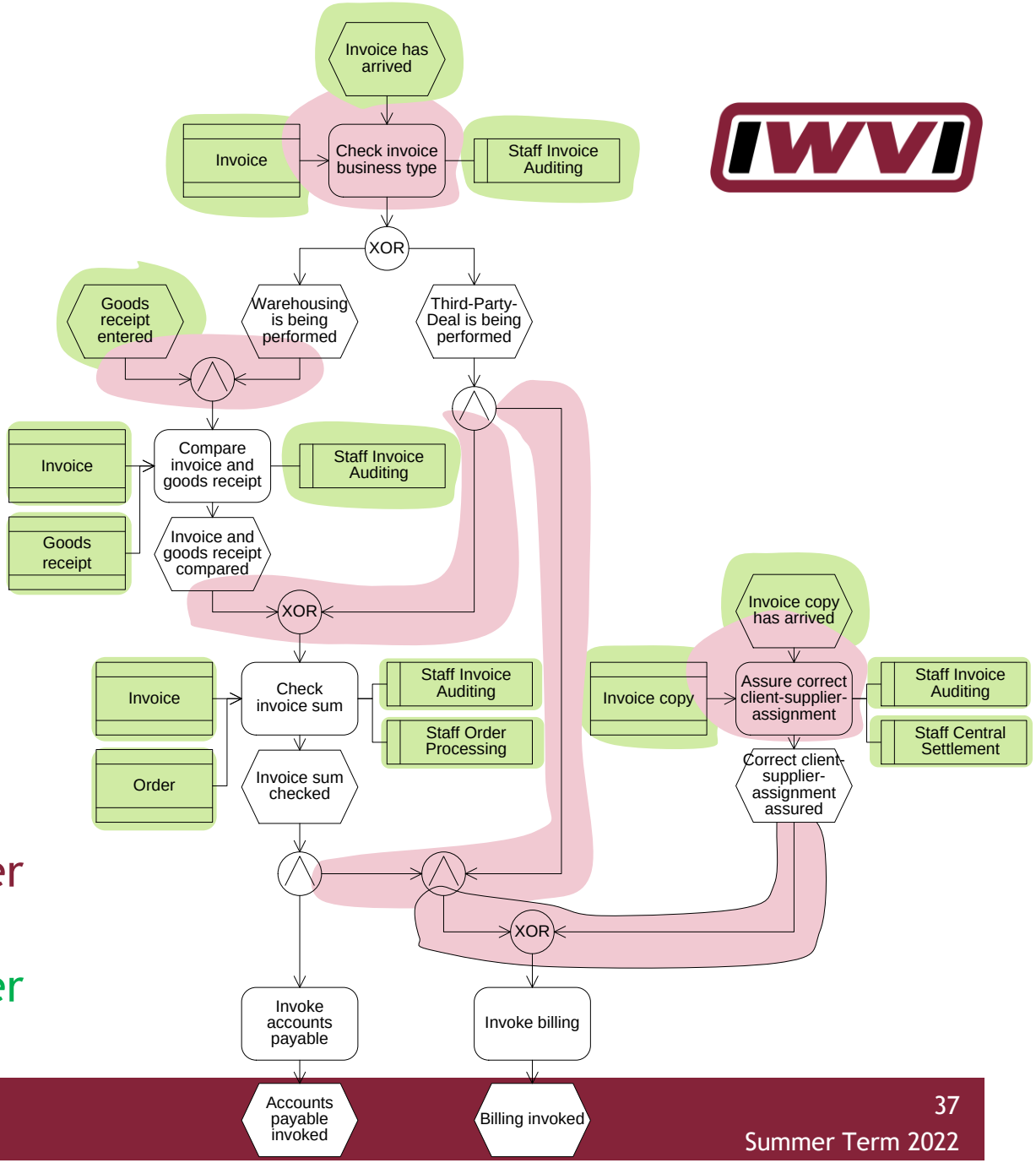


GMQL FUNCTIONS

EWNOPR

- ElementsWithNumberOfPredRelations(X,n_x)
  Returns a set of sets. Each inner set contains an element of *X* and its edges, if it is related to n_x incoming edges of *E*.

- Examples
  - ElementsWithNumberOfPredRelations(V,2)
  - ElementsWithNumberOfPredRelations(V,0)



GMQL

2ND CLASS OF FUNCTIONS: ELEMENTS AND THEIR RELATIONSHIPS

- $ElementsWithNumberOfRelationsOfType(X,t_E,n_x)\subseteq P(Z)$
  Returns a set of sets. Each inner set contains an element of X and its n_x edges of E that are of type t_E.
  $ElWiNuReTy(x,t_E)=\{e\in E\mid (t_E,e)\in I\wedge((x,e)\in D\vee(x,e)\in S)\}\cup\{x\}$
  $ElementsWithNumberOfRelationsOfType(X,t_E,n_x)$
  $=\{ElWiNuReTy(x,t_E)\mid \mid ElWiNuReTy(x,t_E)\mid=n_x+1\}$

- $ElementsWithNumberOfSuccRelationsOfType(X,t_E,n_x)\subseteq P(Z)$
  Returns a set of sets. Each inner set contains an element of X and its n_x outgoing edges of E that are of type t_E.
  $ElWiNuSuReTy(x,t_E)=\{e\in E_d\mid (t_E,r)\in I\wedge(x,e)\in S_d\}\cup\{x\}$
  $ElementsWithNumberOfSuccRelationsOfType(X,t_E,n_x)$
  $=\{ElWiNuSuReTy(x,t_E)\mid \mid ElWiNuSuReTy(x,t_E)\mid=n_x+1\}$

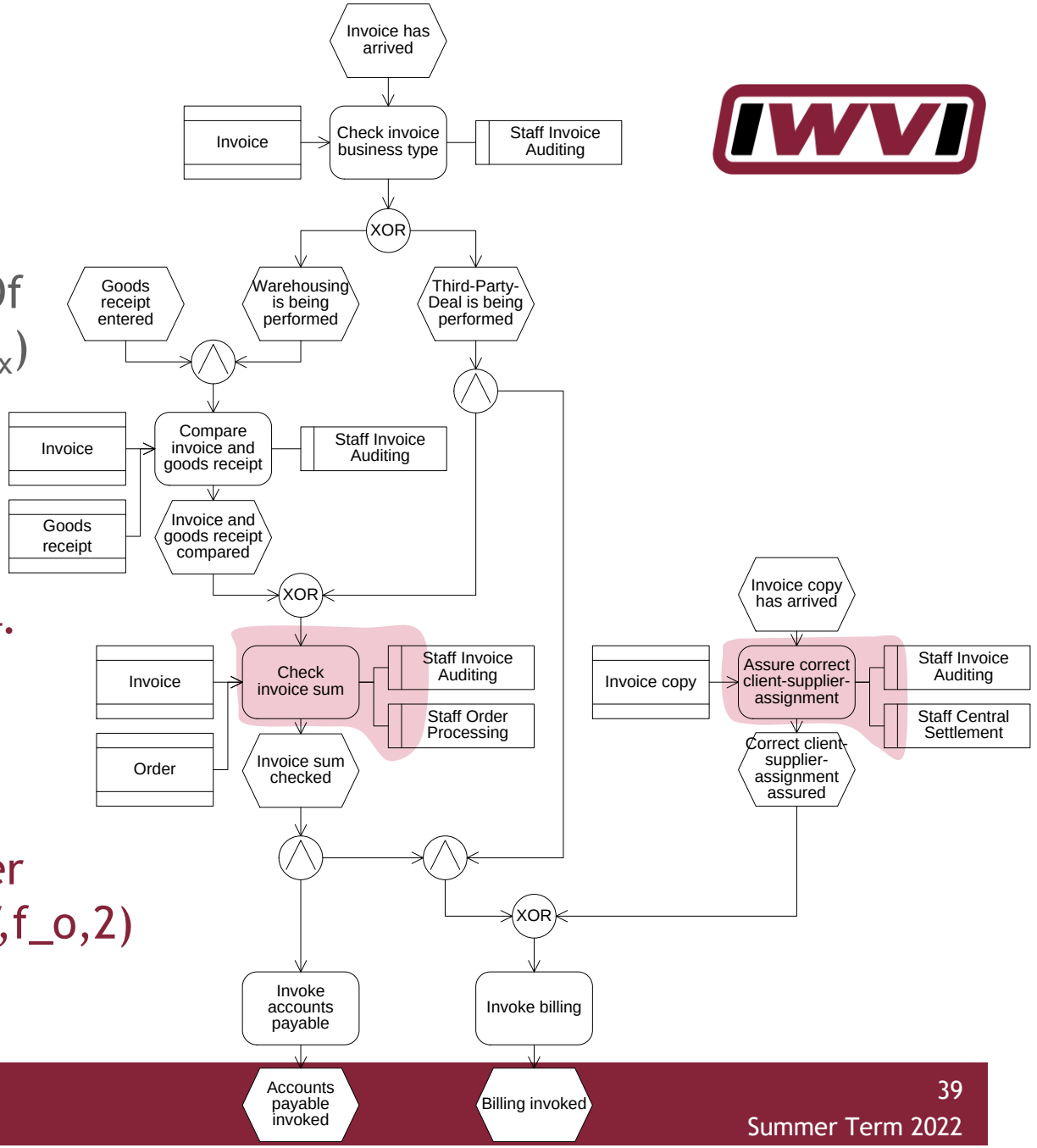
- $ElementsWithNumberOfPredRelationsOfType(X,t_E,n_x)\subseteq P(Z)$
  Returns a set of sets. Each inner set contains an element of X and its n_x incoming edges of E that are of type t_E.
  $ElWiNuPrReTy(x,t_E)=\{e\in E_d\mid (t_E,e)\in I\wedge(x,e)\in D_d\}\cup\{x\}$
  $ElementsWithNumberOfPredRelationsOfType(X,t_E,n_x)$
  $=\{ElWiNuPrReTy(x,t_E)\mid \mid ElWiNuPrReTy(x,t_E)\mid=n_x+1\}$

GMQL FUNCTIONS

EWNOROT

- ElementsWithNumberOfRelationsOfType(X,t_E,n_x)
  Returns a set of sets. Each inner set contains an element of *X* and its n_x edges of *E* that are of type t_E.

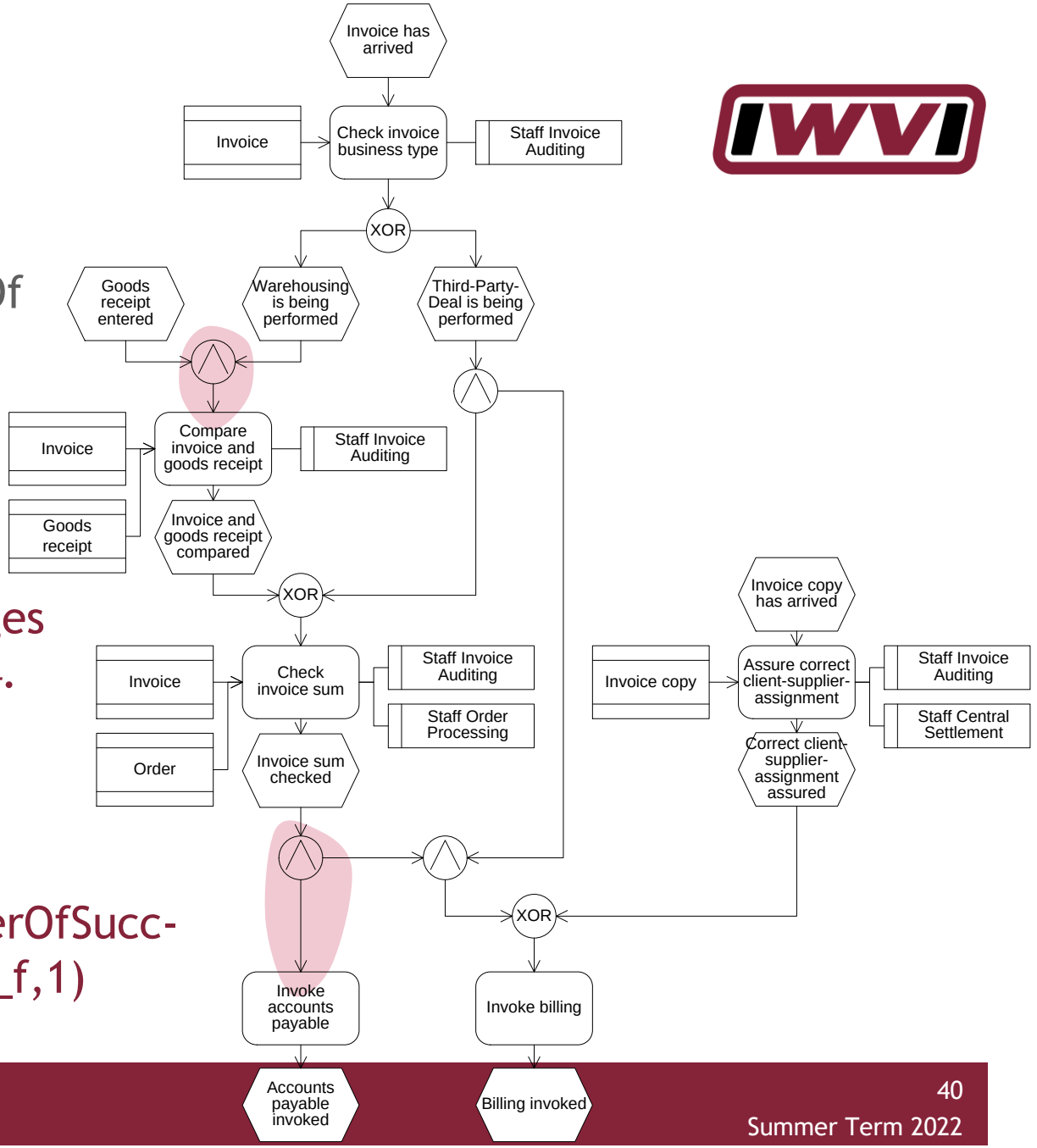
- Example
  - ElementsWithNumberOfRelationsOfType(V,f_o,2)



GMQL FUNCTIONS

EWNOSROT

- ElementsWithNumberOfSuccRelationsOfType(X,t_E,n_x)
  Returns a set of sets. Each inner set contains an element of *X* and its n_x outgoing edges of *E* that are of type t_E.

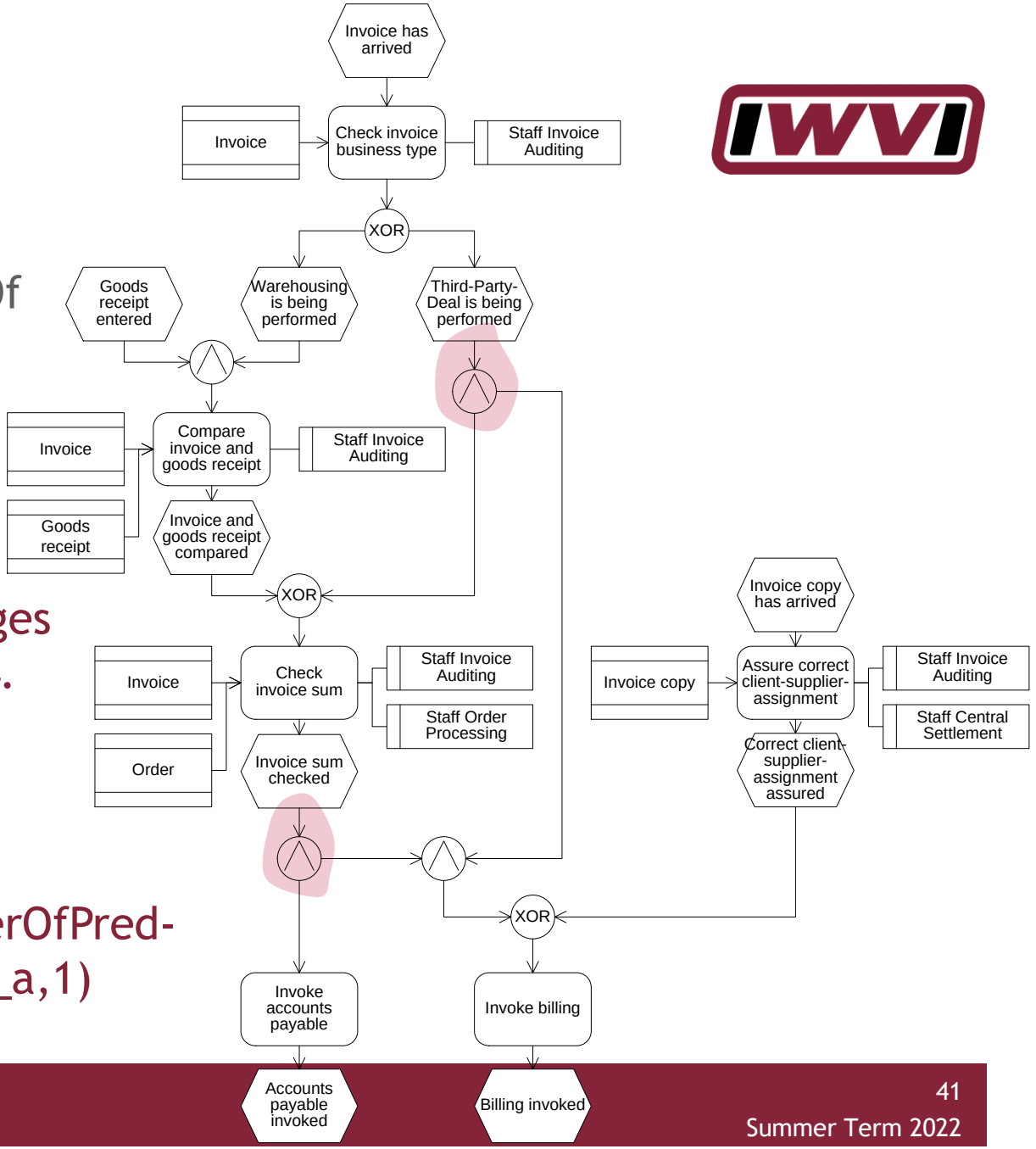


- Example
  - ElementsWithNumberOfSucc-RelationsOfType(V,a_f,1)

GMQL FUNCTIONS

EWNOPROT

- ElementsWithNumberOfPredRelationsOfType(X,t_E,n_x)
  Returns a set of sets. Each inner set contains an element of *X* and its n_x incoming edges of *E* that are of type t_E.



- Example
  - ElementsWithNumberOfPred-RelationsOfType(V,e_a,1)

VARIABLES

- All functions that take non-set parameters as input also accept variables for these parameters
- Variables can thus replace a number, type, attribute, value, or data type parameter
- Variables can be combined to equations (see examples below)

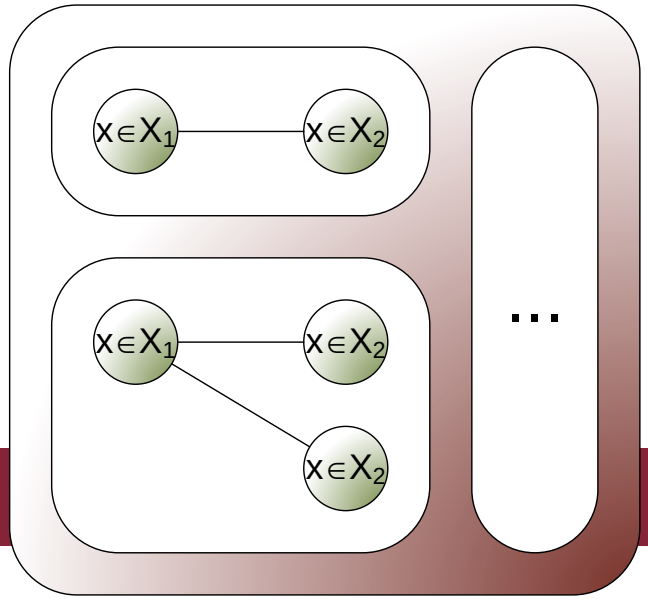
GMQL

3RD CLASS OF FUNCTIONS: ADJACENT ELEMENTS

- *ElementsDirectlyRelated*$(X_1,X_2) \subseteq \mathcal{P}(Z)$
  Returns a set of sets. Each inner set contains an element of X_1, its adjacent elements of X_2, and the connecting edges of E.

  ElDiRe(x_1,X_2)
  $=\{x_2 \in X_2, y_E \in E_u \mid (x_1,y_E) \in S_u \wedge (x_2,y_E) \in D_u \vee (x_2,y_E) \in S_u \wedge (x_1,y_E) \in D_u\} \cup \{x_1\}$

  ElementsDirectlyRelated (X_1,X_2)
  $=\{ElDiRe(x_1,X_2)\},\ x_1 \in X_1$

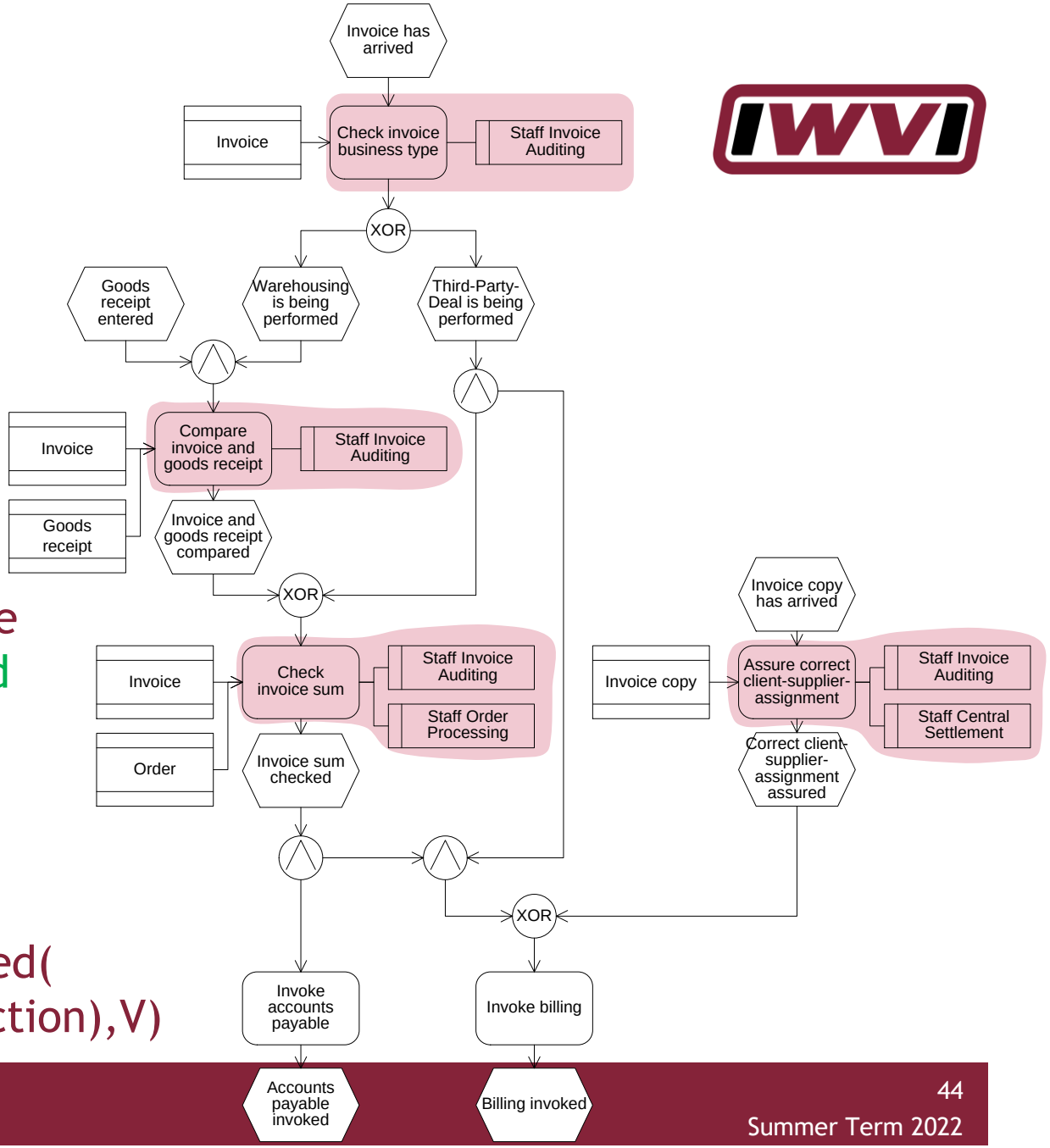


GMQL FUNCTIONS

EDR

- ElementsDirectly Related(X_1,X_2)
  Returns a set of sets. Each inner set contains an element of X_1, its adjacent elements of X_2, and the connecting, undirected edges of E.

- Example
  ElementsDirectlyRelated(ElementsOfType(V,function),V)



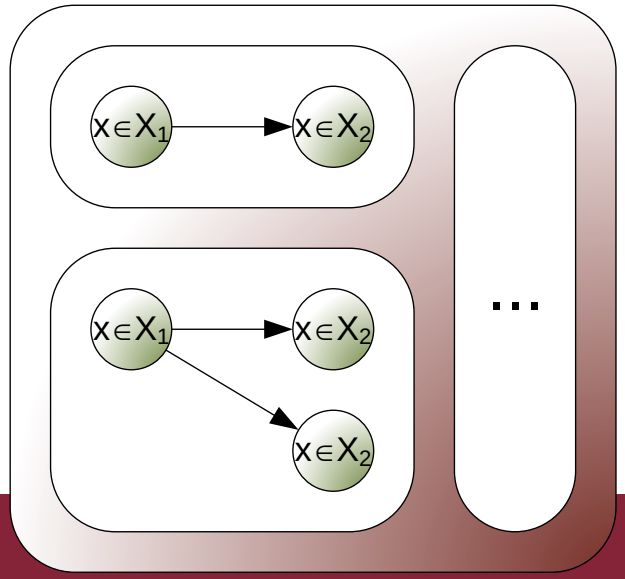
GMQL

3RD CLASS OF FUNCTIONS: ADJACENT ELEMENTS

- $AdjacentSuccessors(X_1,X_2) \subseteq P(Z)$
  Returns a set of sets. Each inner set contains an element of X_1, its adjacent elements of X_2, and the connecting directed edges of E.

  $AdSu(x_1,X_2)=\{x_2 \in X_2, y_E \in E_d \mid (x_1,y_E) \in S_d \wedge (x_2,y_E) \in D_d\} \cup \{x_1\}$

  $AdjacentSuccessors(X_1,X_2)=\{AdSu(x_1,X_2)\},\ x_1 \in X_1$

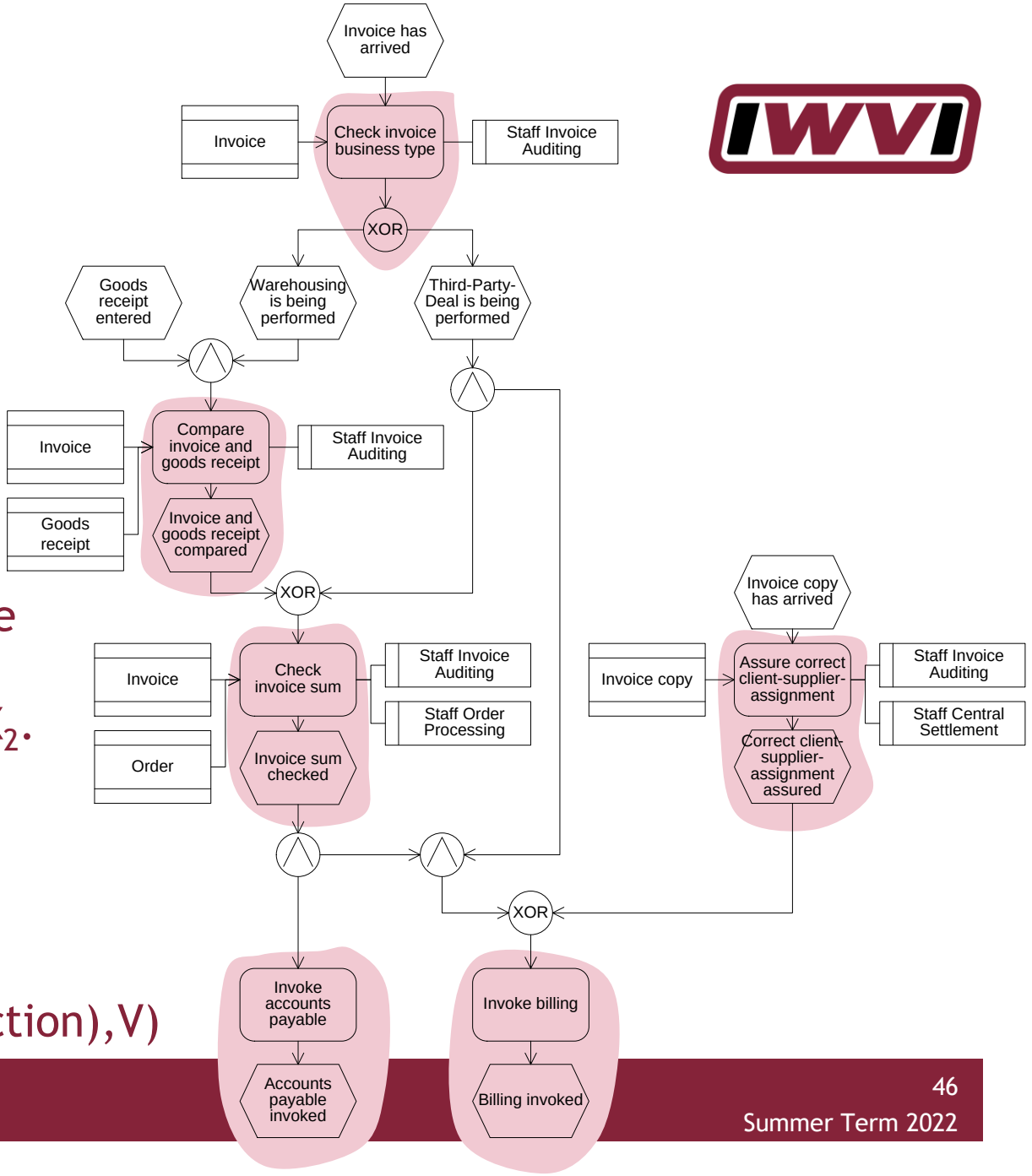


GMQL FUNCTIONS

AS

- Adjacent Successors(X_1,X_2)
  Returns a set of sets. Each inner set contains an element of X_1, its adjacent elements of X_2, and the connecting, directed edges of E from X_1 to X_2.

- Example
  AdjacentSuccessors(ElementsOfType(V,function),V)



GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- Paths functions handle model element sequences of any length (e.g., the path from an EPC start event to an end event)

- Mathematical: sequences or n-tuples
- A help function is needed that converts sequences into sets

- $Set((x_i))=\{x_i \in Z \mid x_i \in (x_i)\} \subseteq Z$

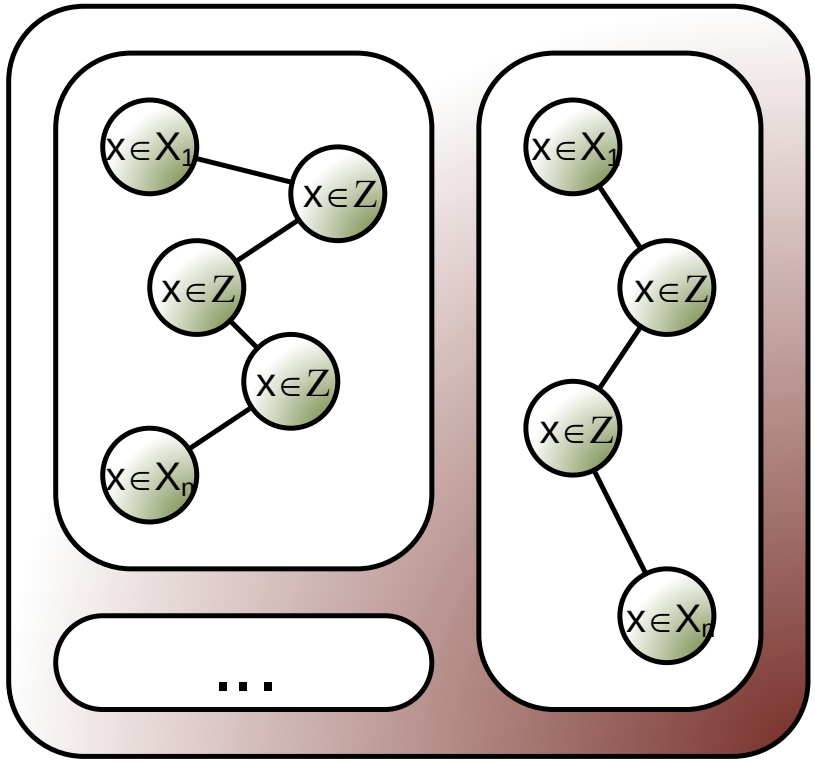
GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- $Paths(X_1,X_n) \subseteq P(Z)$
  Returns a set of sets. Each inner set contains an undirected sequence, which leads from one element of X_1 to one element of X_n.

  $Pa(x_1,x_n) = \{Set((x_1,x_2,\ldots,x_n)) \mid x_2,\ldots,x_{n-1} \in Z \wedge ((x_i,x_{i+1}) \in S_u \vee (x_i,x_{i+1}) \in D_u)) \; \forall 1 \leq i < n\}$

  $Paths(X_1,X_n) = U_{x_1 \in X_1, x_n \in X_n} Pa(x_1,x_n)$

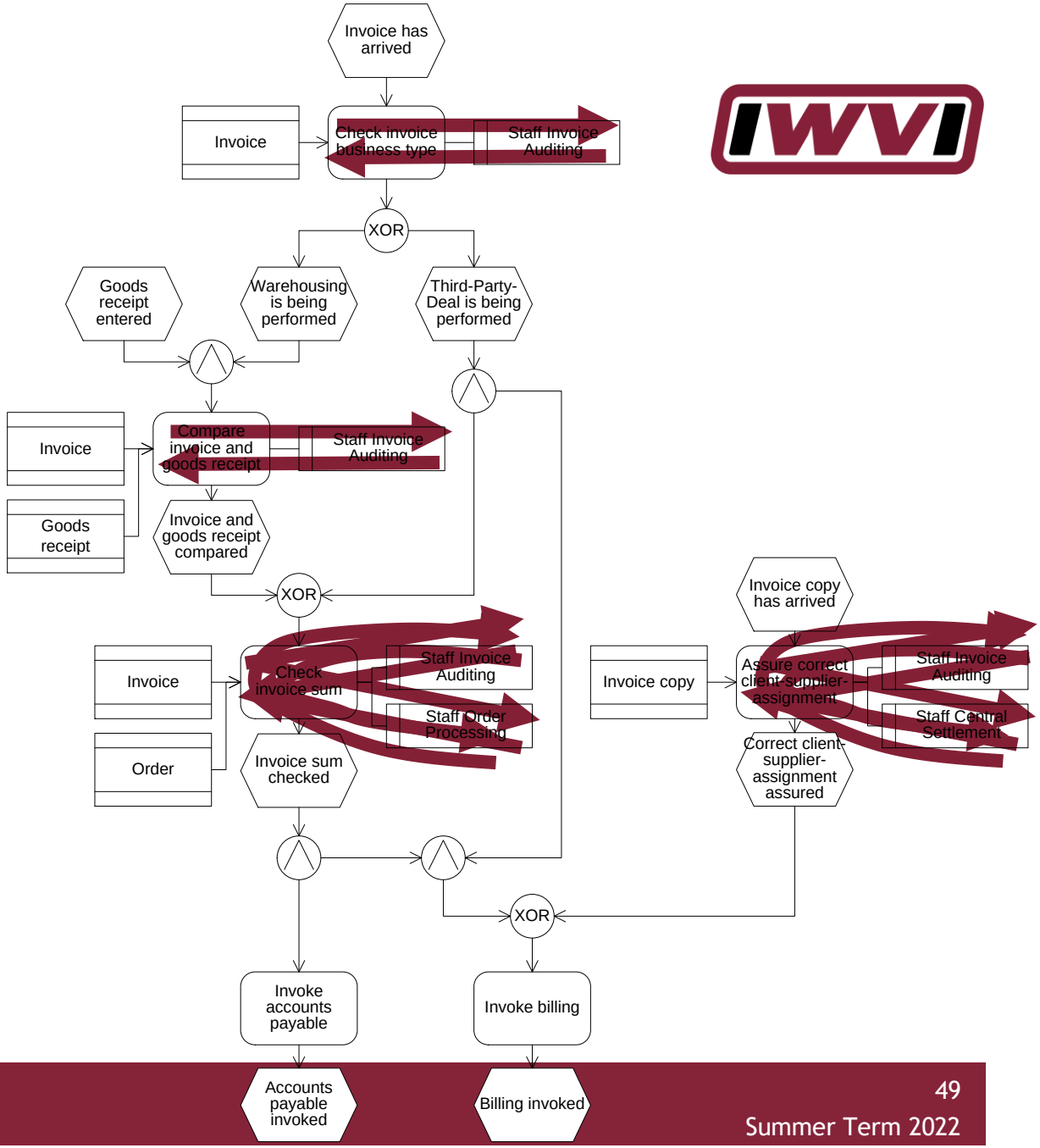


GMQL FUNCTIONS

P

- Paths(X_1,X_n)
  Returns a set of sets. Each inner set contains an undirected sequence, which leads from one element of X_1 to one element of X_n.

- Examples
  - Paths(V,V)



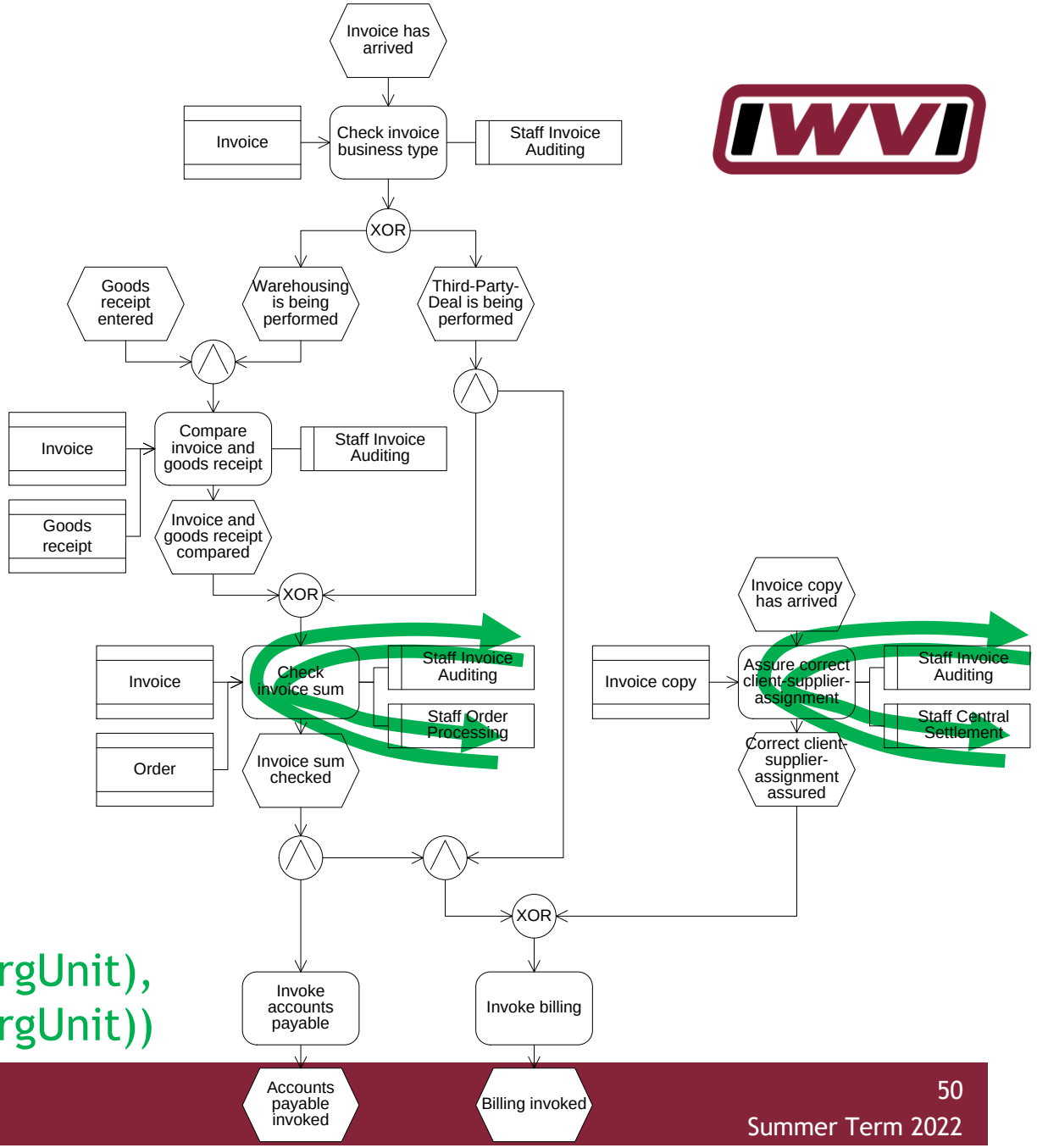
GMQL FUNCTIONS

P

- Paths(X_1,X_n)
  Returns a set of sets. Each inner set contains an undirected sequence, which leads from one element of X_1 to one element of X_n.

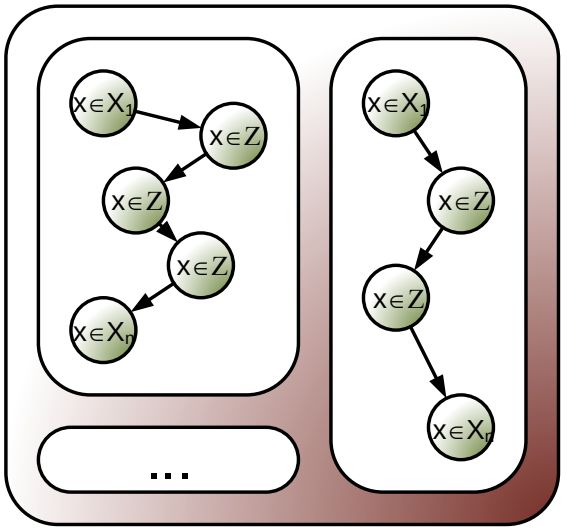
- Examples

  - Paths(ElementsOfType(V,OrgUnit), ElementsOfType(V,OrgUnit))



GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS



- *DirectedPaths*$(X_1,X_n)\subseteq P(Z)$
  Returns a set of sets. Each inner set contains a directed sequence of the same direction, which leads from one element of X_1 to one element of X_n.

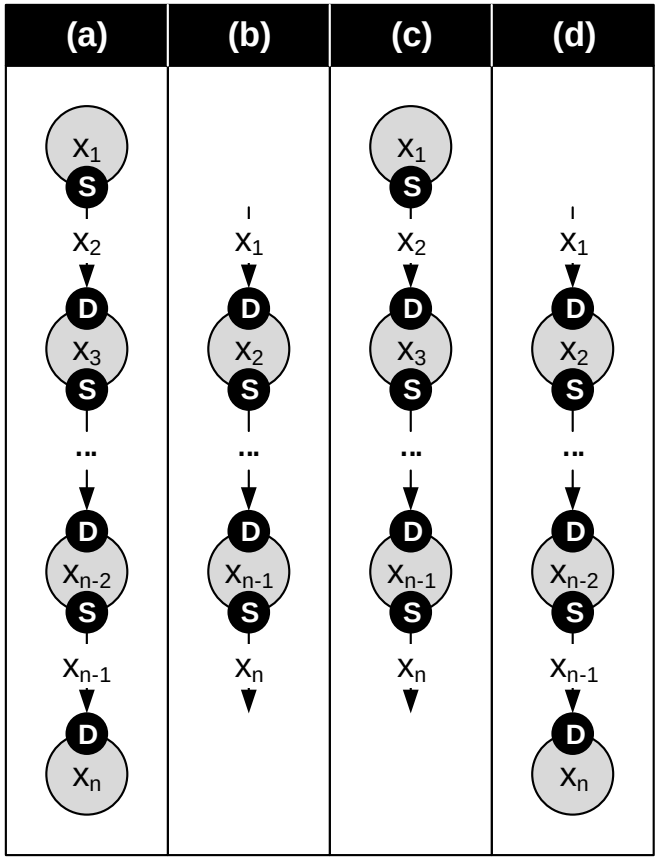
$$DiPa(x_1,x_n)=\{Set((x_1,x_2,\ldots,x_n))\mid$$
$$x_2,\ldots,x_{n-1}\in Z \wedge(((x_{2i-1},x_{2i})\in S_d\wedge(x_{2i+1},x_{2i})\in D_d\ \forall 1\le i\le \lfloor n/2 \rfloor)$$
$$\vee((x_{2i},x_{2i-1})\in D_d\wedge(x_{2i},x_{2i+1})\in S_d\ \forall 1\le i\le \lfloor n/2 \rfloor)$$
$$\vee((x_{2i-1},x_{2i})\in S_d\wedge(x_{2i+1},x_{2i})\in D_d\wedge(x_{n-1},x_n)\in S_d\ \forall 1\le i\le \lfloor n/2 \rfloor-1)$$
$$\vee((x_{2i},x_{2i-1})\in D_d\wedge(x_{2i},x_{2i+1})\in S_d\wedge(x_n,x_{n-1})\in D_d\ \forall 1\le i\le \lfloor n/2 \rfloor-1))\ \ \forall 1\le i<n\}$$

$$\text{DirectedPaths}(X_1,X_n) = U_{x_1\in X_1,x_n\in X_n}\ DiPa(x_1,x_n)$$

GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- Four cases need to be distinguished
  a) Path starts and ends with a vertex
  b) Path starts and ends with an edge
  c) Path starts with a vertex and ends with an edge
  d) Path starts with an edge and ends with a vertex

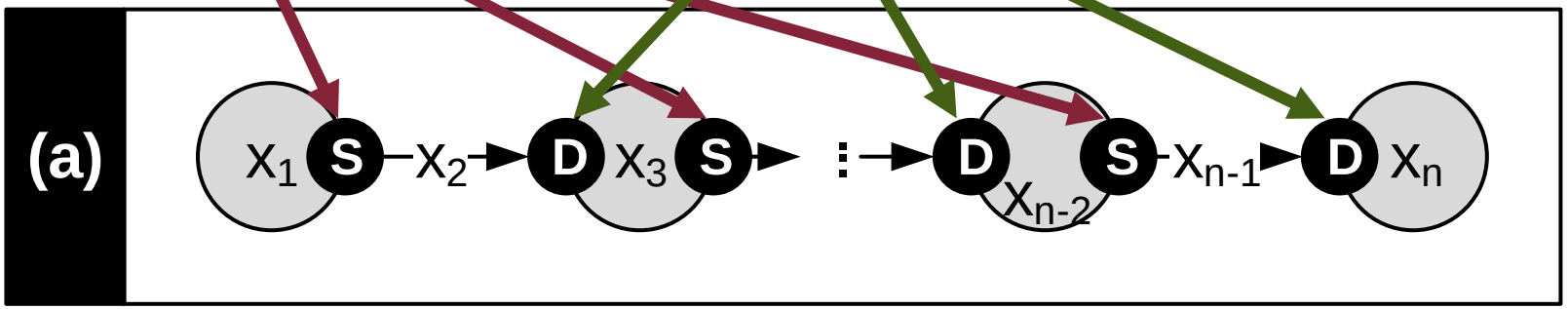


GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- The "$\wedge$" of the formula implies that always a pair of source and destination is evaluated. This means that there has to be an even number of destinations. This, in turn, means that there has to be an odd number of elements on the path.

$$((x_{2i-1}, x_{2i}) \in S_d \wedge (x_{2i+1}, x_{2i}) \in D_d \quad \forall 1 \leq i \leq \lfloor n/2 \rfloor)$$

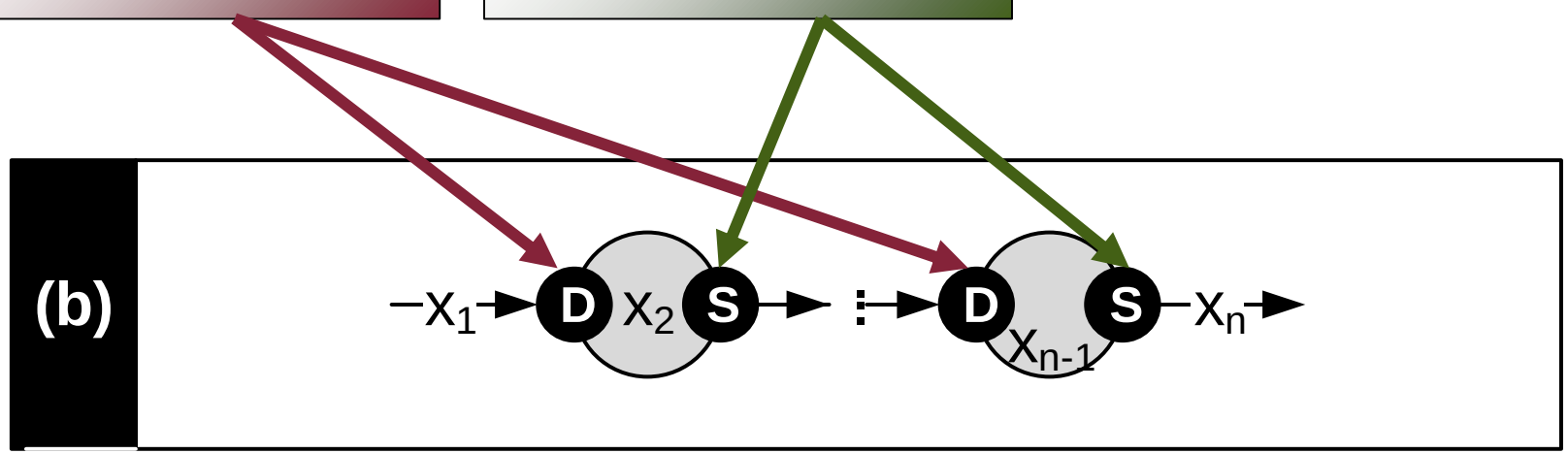


GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- The "$\wedge$" of the formula implies that always a pair of source and destination is evaluated. This means that there has to be an even number of destinations. This, in turn, means that there has to be an odd number of elements on the path.

$$\left(\left(x_{2i}, x_{2i-1}\right) \in D_d \wedge \left(x_{2i}, x_{2i+1}\right) \in S_d \right) \forall 1 \leq i \leq \lfloor n/2 \rfloor$$

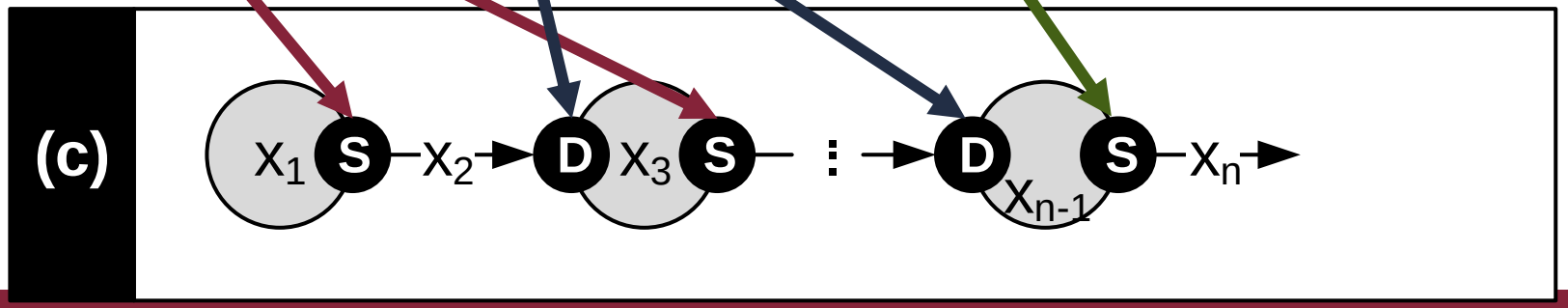


GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- The "∧" of the formula implies that always a pair of source and destination is evaluated. In addition, the equation evaluates the last source (second "∧"). This means that there has to be an odd number of sources/destinations. This, in turn, means that there has to be an even number of elements on the path.

$$((x_{2i-1}, x_{2i}) \in S_d \wedge (x_{2i+1}, x_{2i}) \in D_d \wedge (x_{n-1}, x_n) \in S_d \; \forall 1 \leq i \leq \lfloor n/2 \rfloor - 1)$$

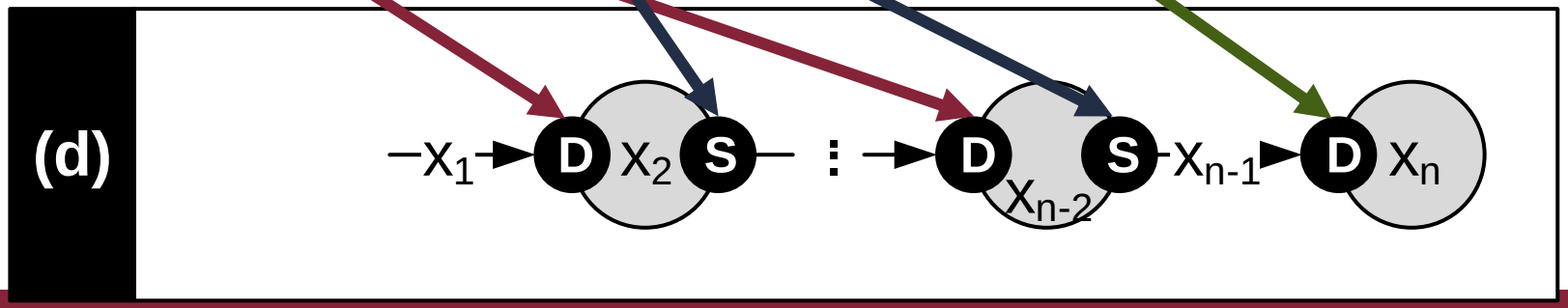


GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- The "$\wedge$" of the formula implies that always a pair of source and destination is evaluated. In addition, the equation evaluates the last destination (second "$\wedge$"). This means that there has to be an odd number of sources/destinations. This, in turn, means that there has to be an even number of elements on the path.

$$((x_{2i}, x_{2i-1}) \in D_d \wedge (x_{2i}, x_{2i+1}) \in S_d \wedge (x_n, x_{n-1}) \in D_d \; \forall 1 \leq i \leq \lfloor n/2 \rfloor - 1)$$

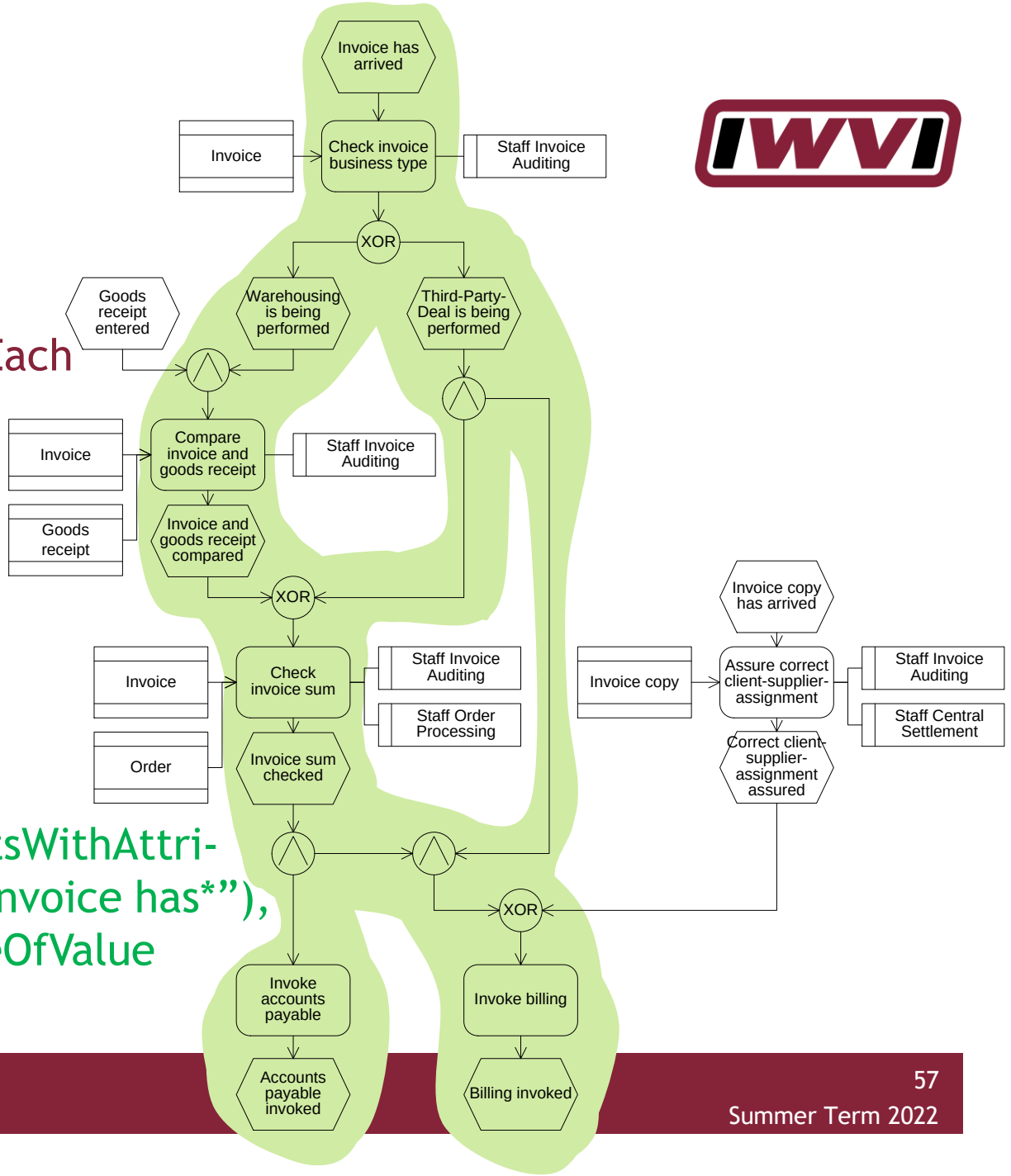


GMQL FUNCTIONS

DP

- DirectedPaths(X_1,X_n)
  Returns a set of sets. Each inner set contains a directed sequence of the same direction, which leads from one element of X_1 to one element of X_n.

- Example
  DirectedPaths(ElementsWithAttri-buteOfValue(V,label,"Invoice has*"), ElementsWithAttributeOfValue (V,label,"*invoked"))



GMQL

4TH CLASS OF FUNCTIONS: PATHS OF ELEMENTS

- *PathsContainingElements*$(X_1,X_n,X_c) \subseteq \mathcal{P}(Z)$
  Returns a set of sets. Each inner set contains one undirected path from one element of X_1 to one element of X_n. At least one element of X_c has to be contained on that path.

$$PaCoEl(x_1,x_n,X_c)=\{Set((x_1,x_2,\ldots,x_n)) \mid x_2,\ldots,x_{n-1} \in Z \wedge \exists x_c \in \{x_2,\ldots,x_{n-1}\} \wedge ((x_i,x_{i+1}) \in S_u \vee (x_i,x_{i+1}) \in D_u) \; \forall 1 \leq i < n\}$$

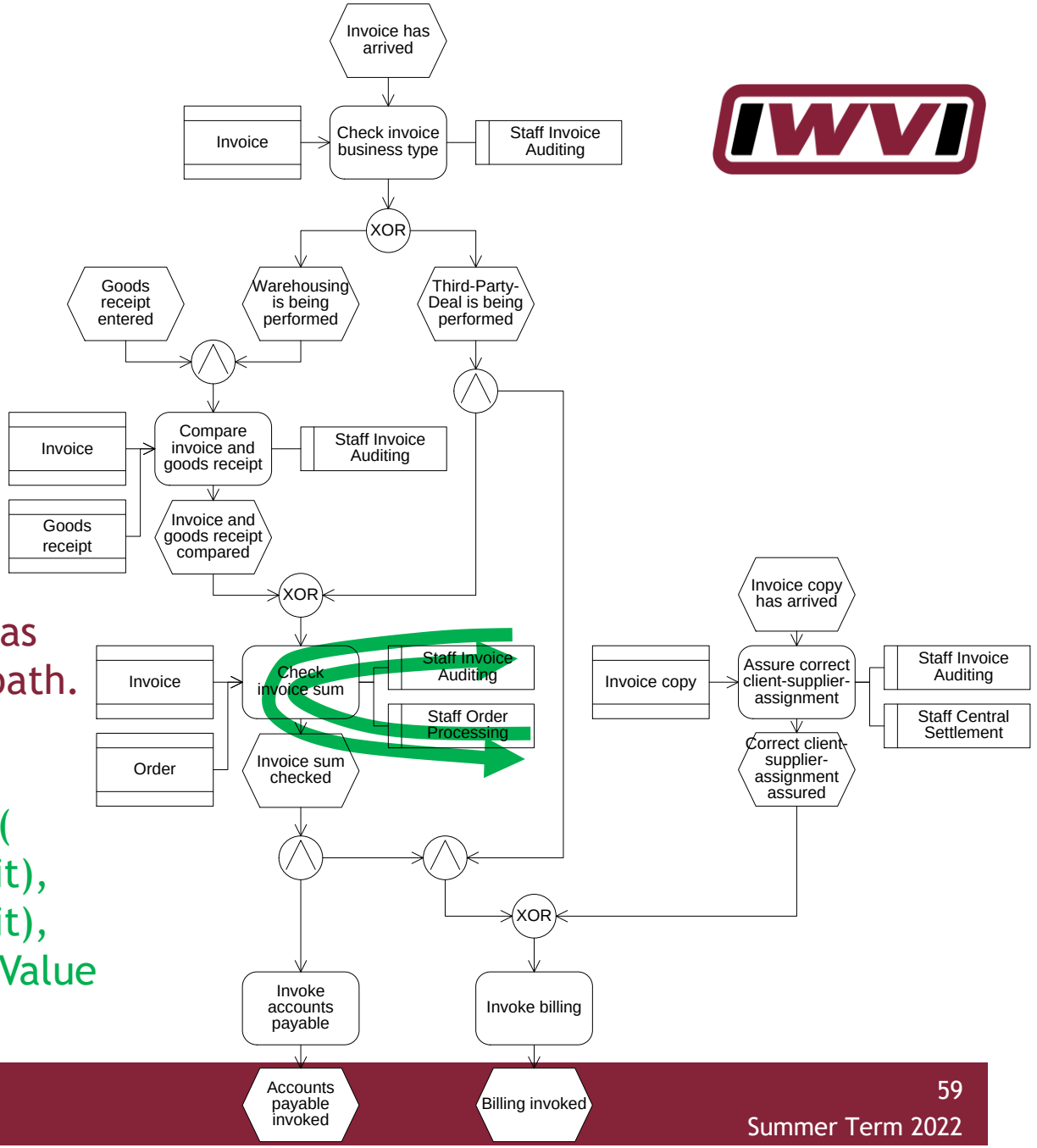
$$\text{PathsContainingElements}(X_1,X_n,X_c) = \bigcup_{x_1 \in X_1, x_n \in X_n} \text{PaCoEl}(x_1,x_n,X_c)$$

GMQL FUNCTIONS

PCE

- PathsContaining Elements(X_1,X_n,X_c)
  Returns a set of sets. Each inner set contains one undirected path from one element of X_1 to one element of X_n. At least one element of X_c has to be contained on that path.

- Example
  PathsContainingElements(ElementsOfType(V,OrgUnit), ElementsOfType(V,OrgUnit), ElementsWithAttributeOfValue (V,label,"*invoice*"))



GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- *DirectedPathsContainingElements*$(X_1,X_n,X_c)\subseteq\mathcal{P}(Z)$
  Returns a set of sets. Each inner set contains one directed path from one element of X_1 to one element of X_n. At least one element of X_c has to be contained on that path.

$$DiPaCoEl(x_1,x_n,X_c)=\{Set((x_1,x_2,\ldots,x_n))\mid x_2,\ldots,x_{n-1}\in Z\wedge\exists x_c\in\{x_2,\ldots,x_{n-1}\}\wedge$$
$$(((x_{2i-1},x_{2i})\in S_d\wedge(x_{2i+1},x_{2i})\in D_d\ \forall 1\leq i\leq\lfloor n/2\rfloor)$$
$$\vee((x_{2i},x_{2i-1})\in D_d\wedge(x_{2i},x_{2i+1})\in S_d\ \forall 1\leq i\leq\lfloor n/2\rfloor)$$
$$\vee((x_{2i-1},x_{2i})\in S_d\wedge(x_{2i+1},x_{2i})\in D_d\wedge(x_{n-1},x_n)\in S_d\ \forall 1\leq i\leq\lfloor n/2\rfloor-1)$$
$$\vee((x_{2i},x_{2i-1})\in D_d\wedge(x_{2i},x_{2i+1})\in S_d\wedge(x_n,x_{n-1})\in D_d\ \forall 1\leq i\leq\lfloor n/2\rfloor-1))\ \forall 1\leq i<n\}$$

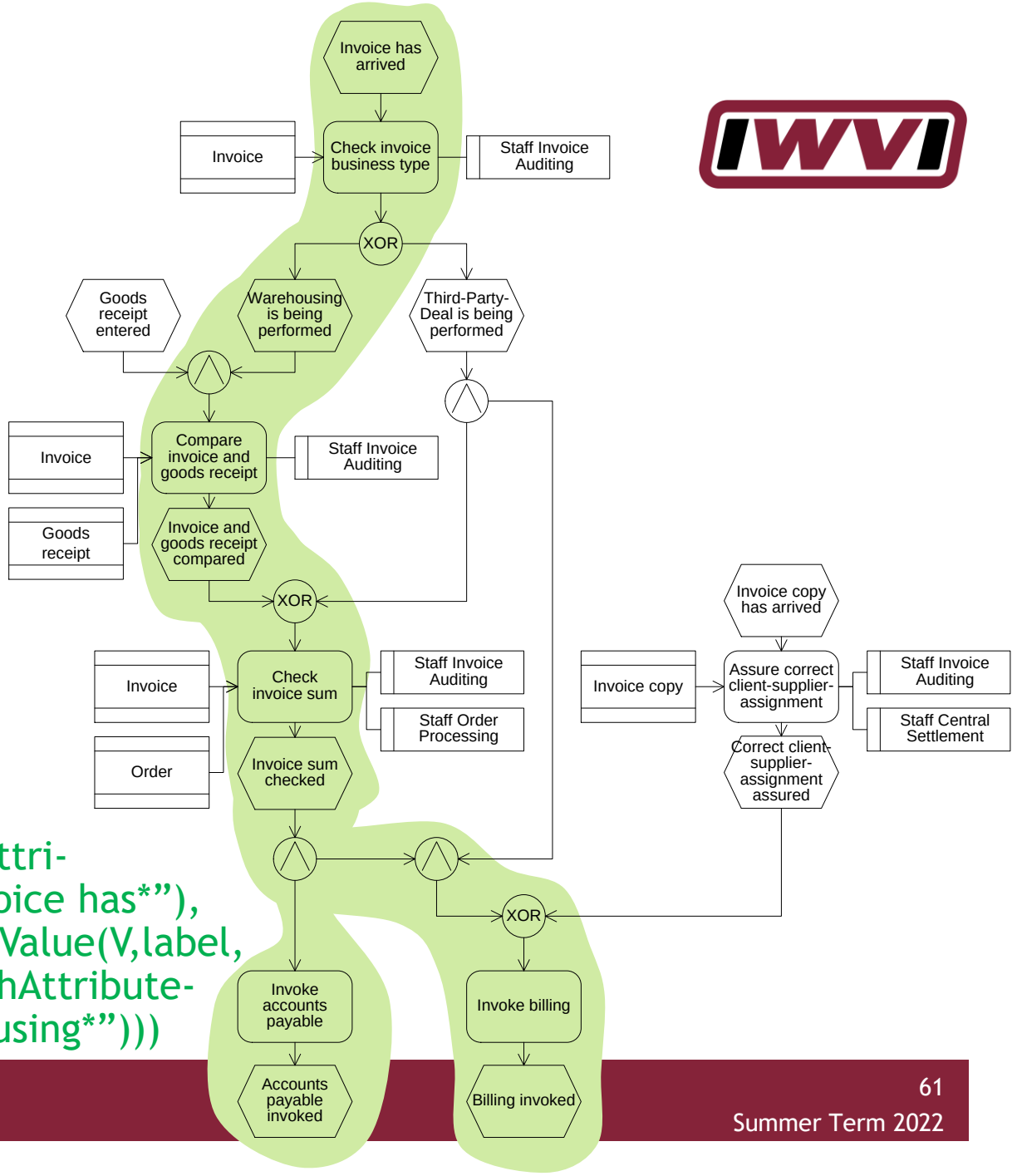
$$PathsContainingElements(X_1,X_n,X_c)=\bigcup_{x_1\in X_1,x_n\in X_n}DiPaCoEl(x_1,x_n,X_c)$$

GMQL FUNCTIONS

DPCE

- DirectedPathsContaining Elements(X_1,X_n,X_c)
  Returns a set of sets.
  Each inner set contains one directed path from one element of X_1 to one element of X_n.
  At least one element of X_c has to be contained on that path.

- Example
  DirectedPathsContaining-Elements(ElementsWithAttri-buteOfValue(V,label,"Invoice has*"), ElementsWithAttributeOfValue(V,label, "*invoked"),ElementsWithAttribute-OfValue(V,label,"Warehousing*")))



GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- $PathsNotContainingElements(X_1,X_n,X_c) \subseteq P(Z)$
  Returns a set of sets. Each inner set contains one undirected path from one element of X_1 to one element of X_n. No element of X_c may be contained on that path.

$$PaNoCoEl(x_1,x_n,X_c)=\{Set((x_1,x_2,\ldots,x_n)) \mid x_2,\ldots,x_{n-1} \in Z \setminus X_c \wedge ((x_i,x_{i+1}) \in S_u \vee (x_i,x_{i+1}) \in D_u) \forall\, 1 \leq i < n\}$$

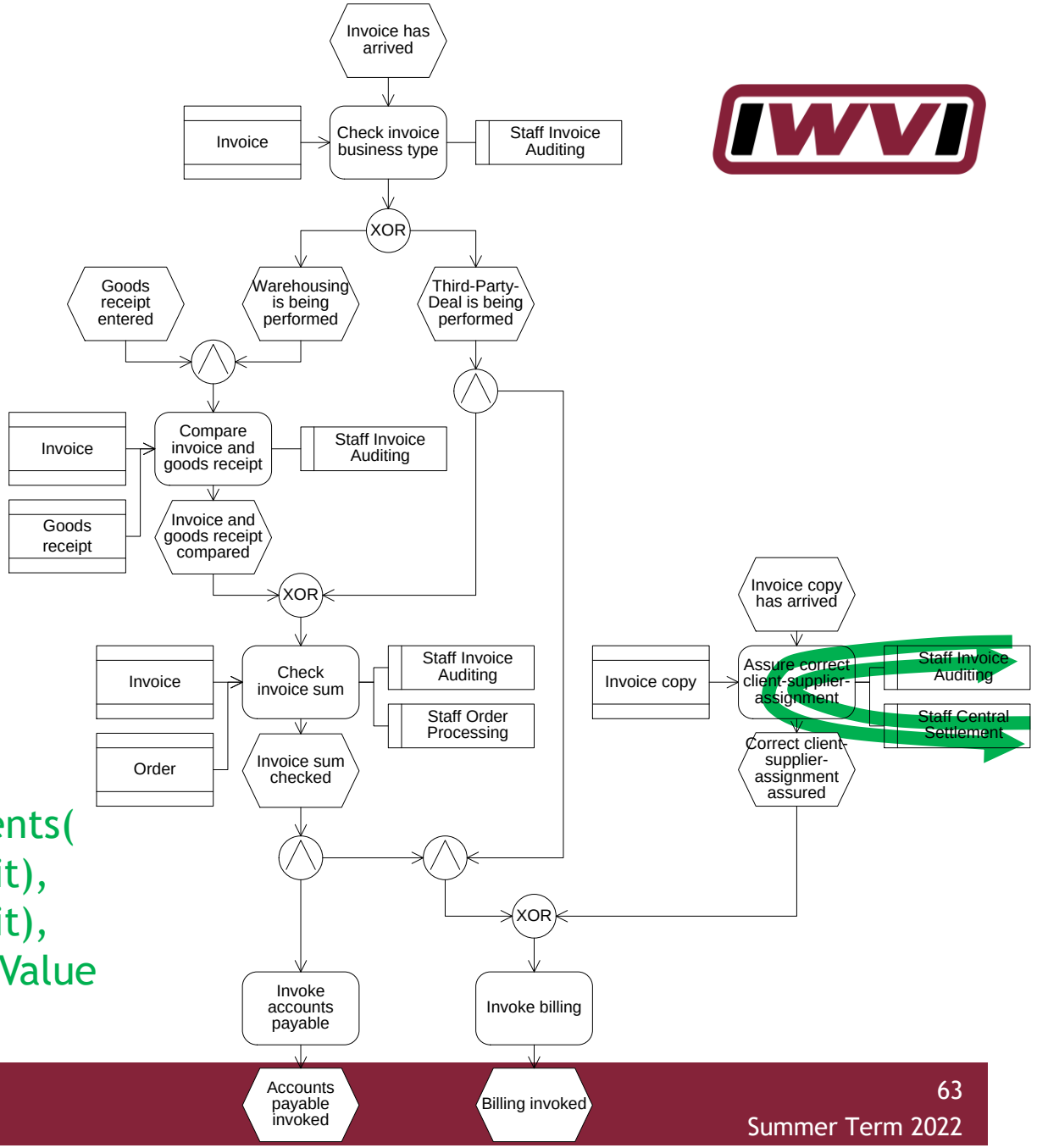
$$PathsNotContainingElements(X_1,X_n,X_c) = U_{x_1 \in X_1, x_n \in X_n} PaNoCoEl(x_1,x_n,X_c)$$

GMQL FUNCTIONS

PNCE

- PathsNotContaining Elements(X_1,X_n,X_c)
  Returns a set of sets. Each inner set contains one undirected path from one element of X_1 to one element of X_n. No element of X_c may be contained on that path.

- Example
  PathsNotContainingElements(ElementsOfType(V,OrgUnit), ElementsOfType(V,OrgUnit), ElementsWithAttributeOfValue (V,label,"*invoice*"))



GMQL

4TH CLASS OF FUNCTIONS: ELEMENT PATHS

- *DirectedPathsNotContainingElements*$(X_1,X_n,X_c)\subseteq P(Z)$
  Returns a set of sets. Each inner set contains one directed path from one element of X_1 to one element of X_n. No element of X_c may be contained on that path.

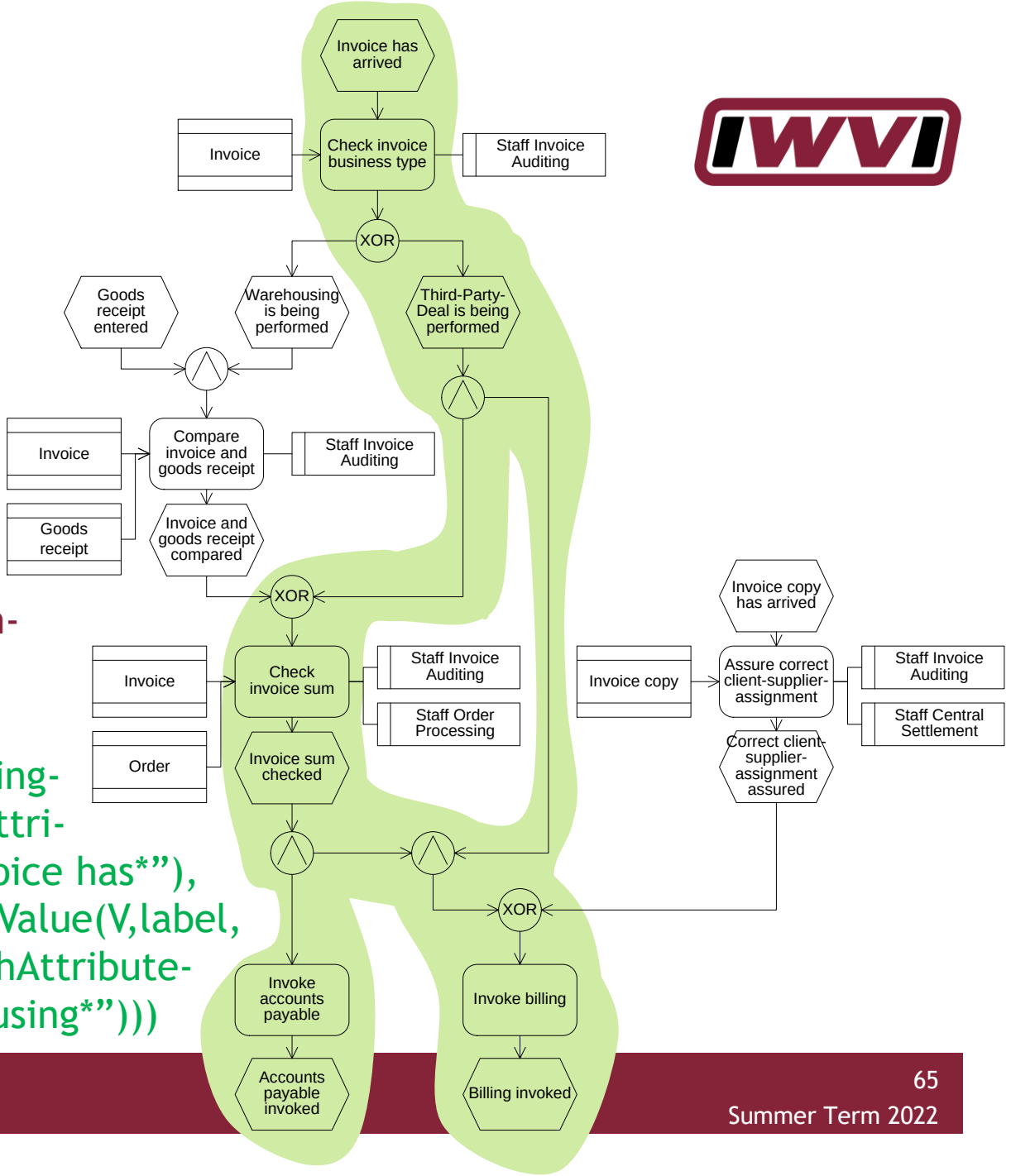
$$DiPaNoCoEl(x_1,x_n,X_c)=\{Set((x_1,x_2,\ldots,x_n)) \mid x_2,\ldots,x_{n-1}\in Z\setminus X_c$$
$$\wedge(((x_{2i-1},x_{2i})\in S_d\wedge(x_{2i+1},x_{2i})\in D_d\ \forall 1\leq i\leq \lfloor n/2 \rfloor)$$
$$\vee((x_{2i},x_{2i-1})\in D_d\wedge(x_{2i},x_{2i+1})\in S_d\ \forall 1\leq i\leq \lfloor n/2 \rfloor)$$
$$\vee((x_{2i-1},x_{2i})\in S_d\wedge(x_{2i+1},x_{2i})\in D_d\wedge(x_{n-1},x_n)\in S_d\ \forall 1\leq i\leq \lfloor n/2 \rfloor -1)$$
$$\vee((x_{2i},x_{2i-1})\in D_d\wedge(x_{2i},x_{2i+1})\in S_d\wedge(x_n,x_{n-1})\in D_d$$
$$\forall 1\leq i\leq \lfloor n/2 \rfloor -1))\ \forall 1\leq i<n\}$$

$$DirectedPathsNotContainingElements(X_1,X_n,X_c) = \bigcup_{x_1\in X_1,x_n\in X_n} DiPaNoCoEl(x_1,x_n,X_c)$$

GMQL FUNCTIONS

DPNCE

- DirectedPathsNotCon-tainingElements(X_1,X_n,X_c)
  Returns a set of sets. Each inner set contains one directed path from one element of X_1 to one element of X_n. No element of X_c may be con-tained on that path.
- Example
  DirectedPathsNotContaining-Elements(ElementsWithAttri-buteOfValue(V,label,"Invoice has*"), ElementsWithAttributeOfValue(V,label, "*invoked"),ElementsWithAttribute-OfValue(V,label,"Warehousing*")))



GMQL FUNCTIONS

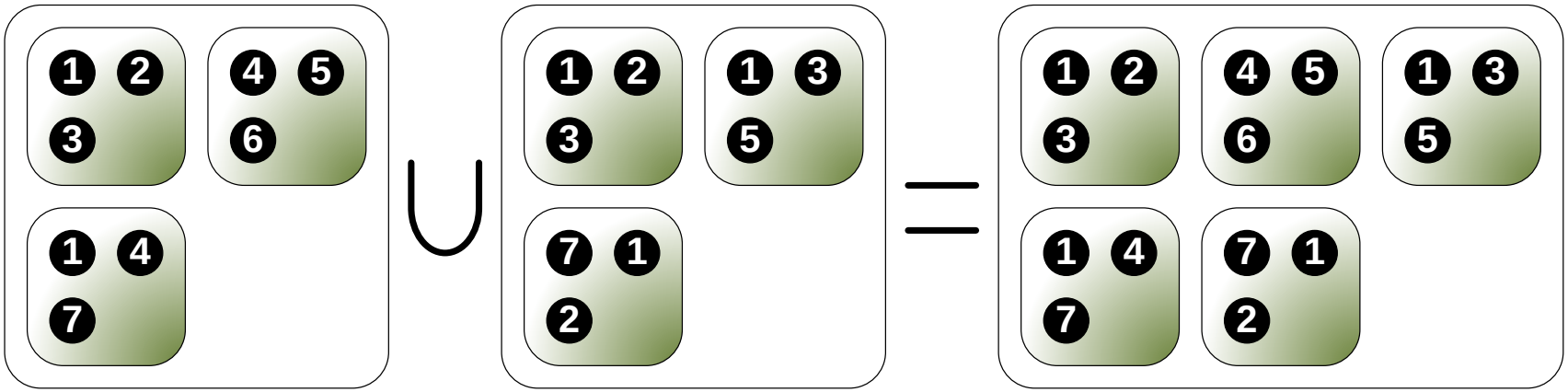
L, DL, LCE, DLCE, LNCE, DLNCE

- The following Loop functions actually do almost the same as their Path function counterparts, however, they are searching for paths that start and end in the same element (which is always out of X)
  - Loops(X)=$U_{x \in X}$ Pa(x,x)
  - DirectedLoops(X)=$U_{x \in X}$ DiPa(x,x)
  - LoopsContainingElements(X,X_c)=$U_{x \in X}$ PaCoEl(x,x,X_c)
  - DirectedLoopsContainingElements(X,X_c)=$U_{x \in X}$ DiPaCoEl(x,x,X_c)
  - LoopsNotContainingElements(X,X_c)=$U_{x \in X}$ PaNoCoEl(x,x,X_c)
  - DirectedLoopsNotContainingElements(X,Xc) =$U_{x \in X}$ DiPaNoCoEl(x,x,X_c)

COMBINATION OF FUNCTION RESULTS

GMQL SET OPERATORS

- The common UNION operator
- Takes either two simple sets or two sets of sets as input

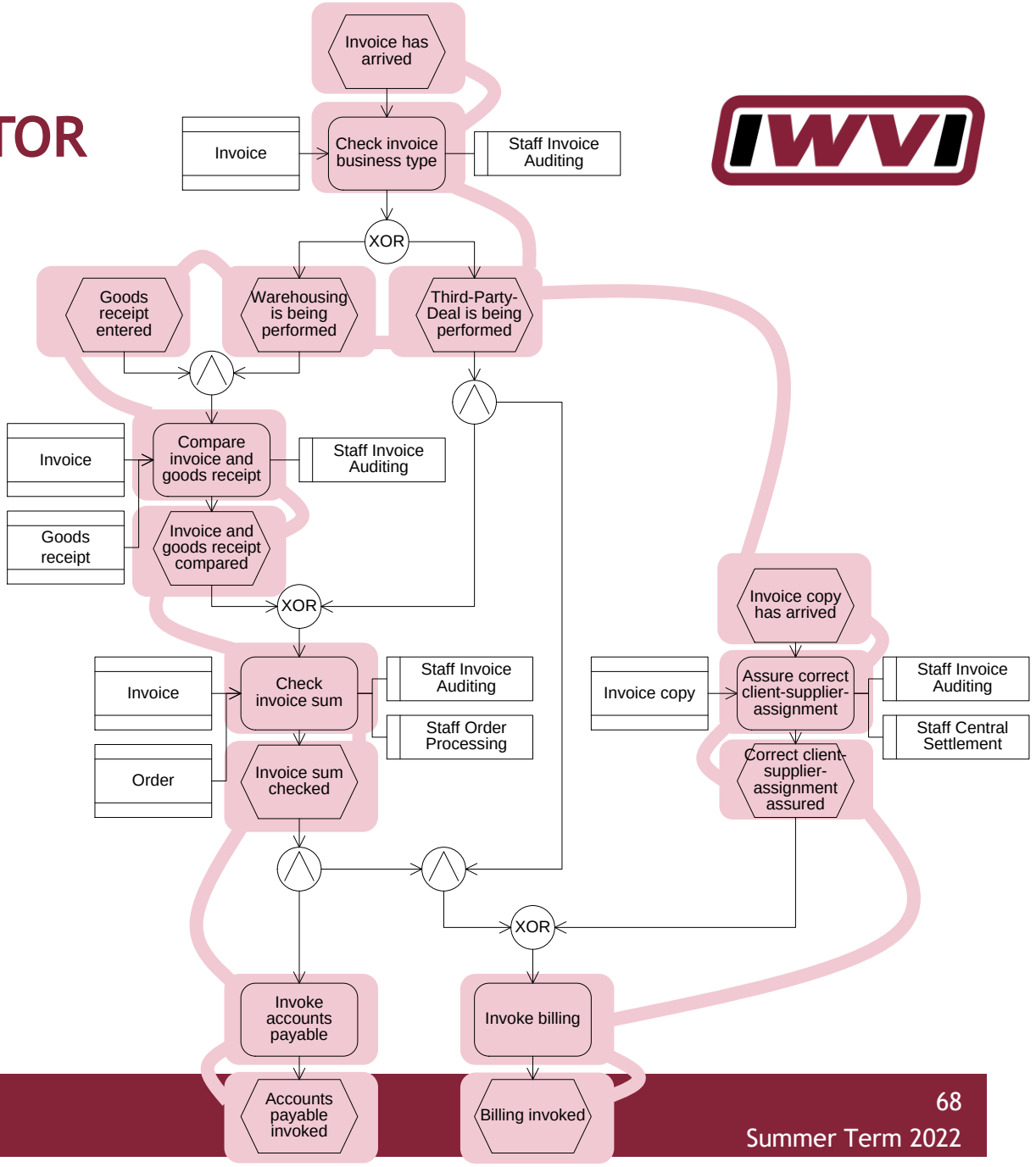


THE UNION OPERATOR

EXAMPLES

- UNION(ElementsOfType(V,function), ElementsOfType(V,event))

- Result: One simple set! (see single outline in the visulaization)

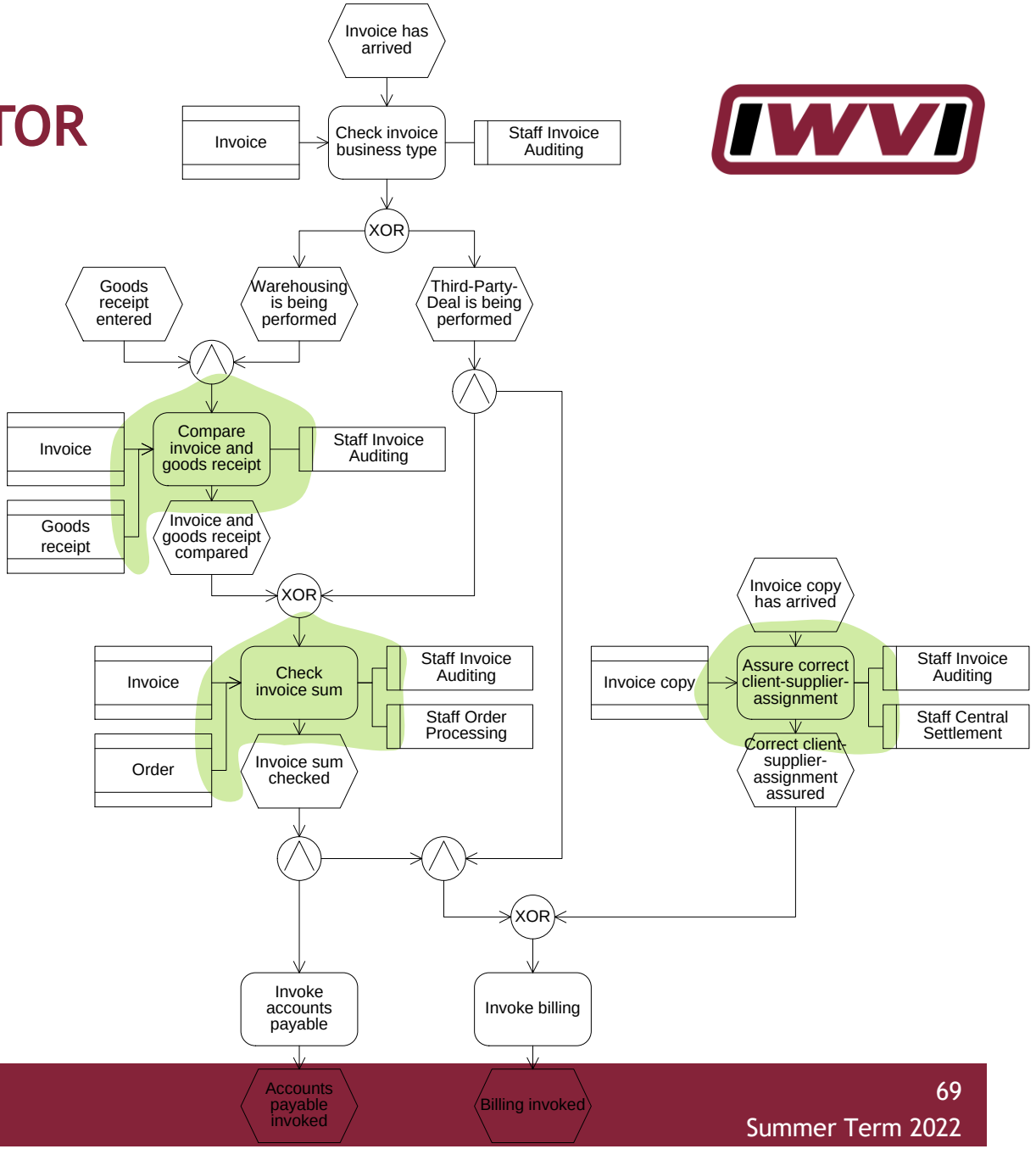


THE UNION OPERATOR

EXAMPLES

- UNION(ElementsWith-NumberOf-Relations(V,5), ElementsWith-NumberOf-Relations(v,6))

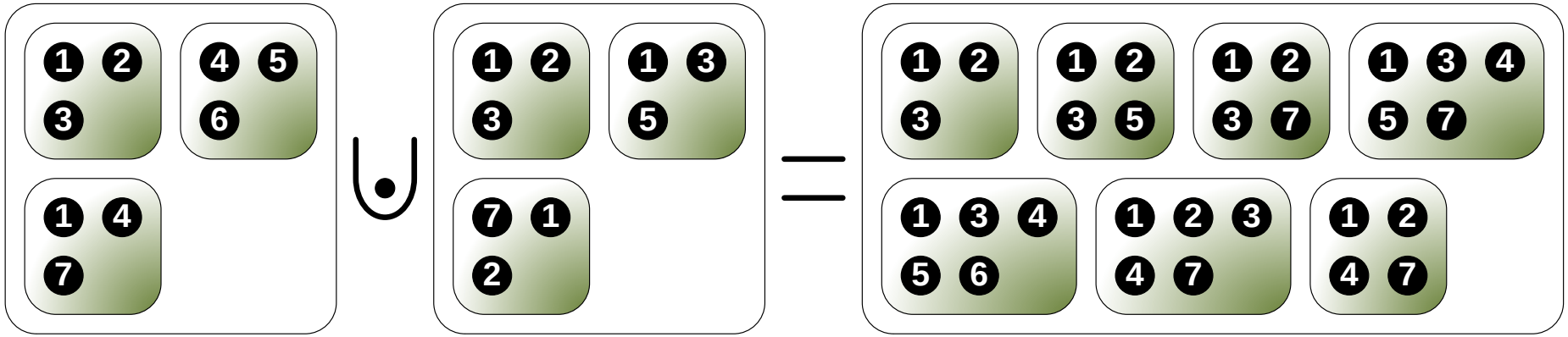
- Result: A set of sets!



COMBINATION OF FUNCTION RESULTS

GMQL SET OPERATORS

- The JOIN operator performs a UNION on two inner sets that have at least one element in common
- Only works on two sets of sets as inputs



THE JOIN OPERATOR

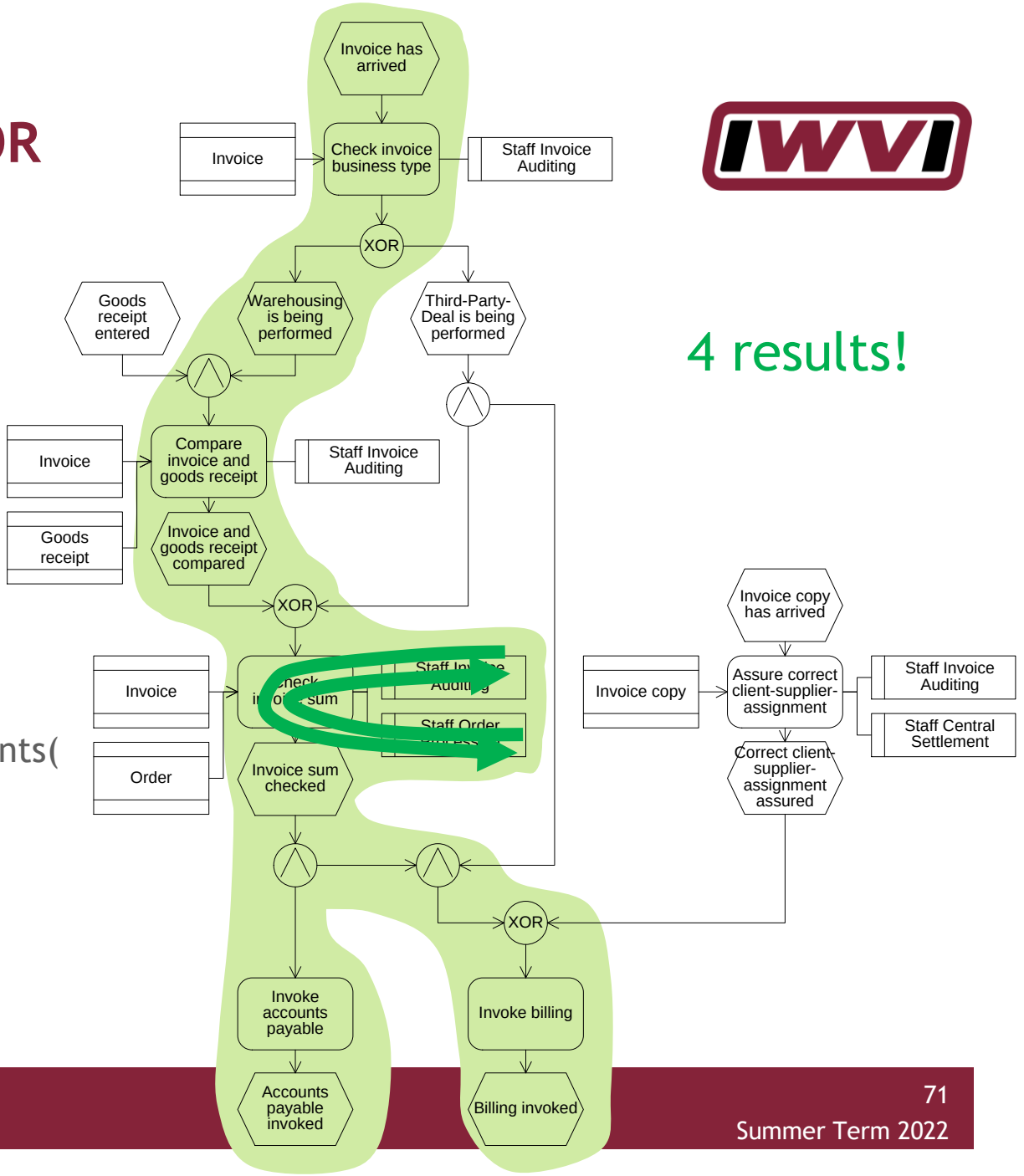
EXAMPLE

- Result: A set of sets!

- JOIN(

  Paths(
  ElementsOfType(V,OrgUnit),
  ElementsOfType(V,OrgUnit)),

  DirectedPathsContainingElements(
  ElementsWithAttributeOfValue
  (V,label,"Invoice has*"),
  ElementsWithAttributeOfValue
  (V,label, "*invoked"),
  ElementsWithAttributeOfValue
  (V,label,"Warehousing*")))

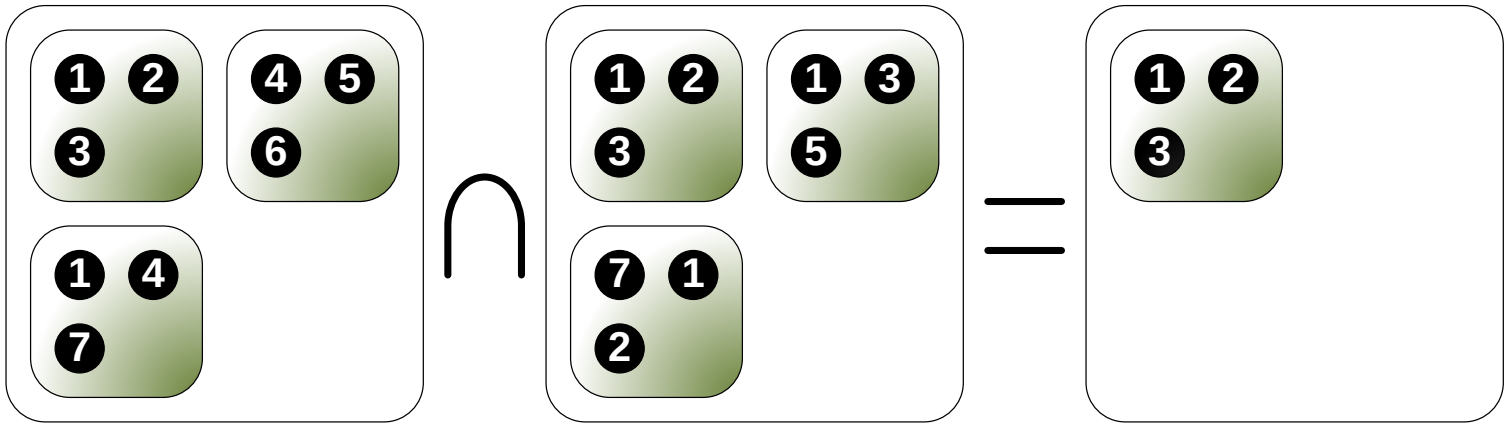


4 results!

COMBINATION OF FUNCTION RESULTS

GMQL SET OPERATORS

- The common INTERSECTION operator
- Takes either two simple sets or two sets of sets as input

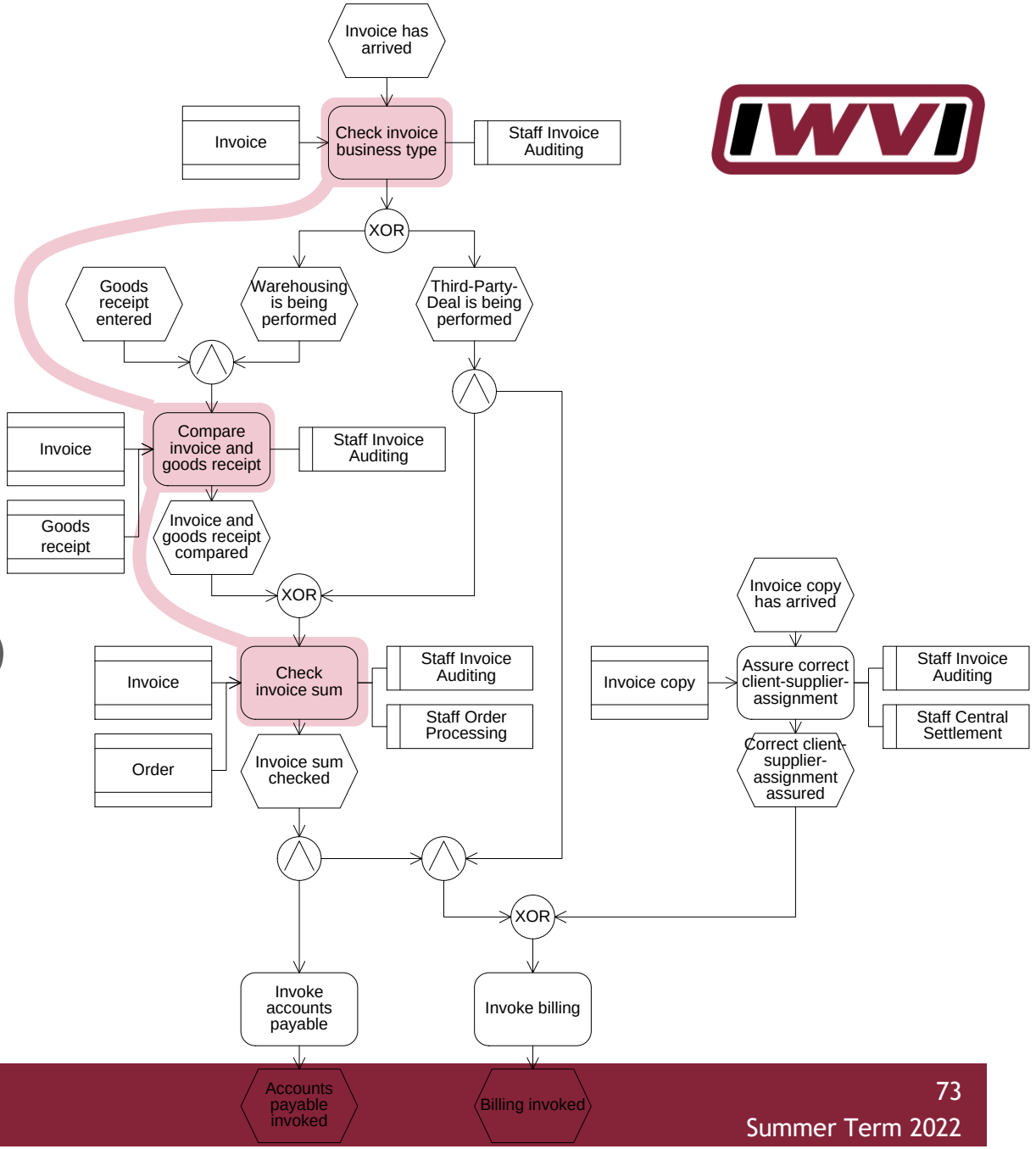


THE INTERSECTION OPERATOR

EXAMPLES

- Intersection(ElementsOfType (V,function), ElementsWith-AttributeOfValue (V,label,“*invoice*“))

- Result: One simple set! (see single outline in the visulaization)



THE INTERSECTION OPERATOR

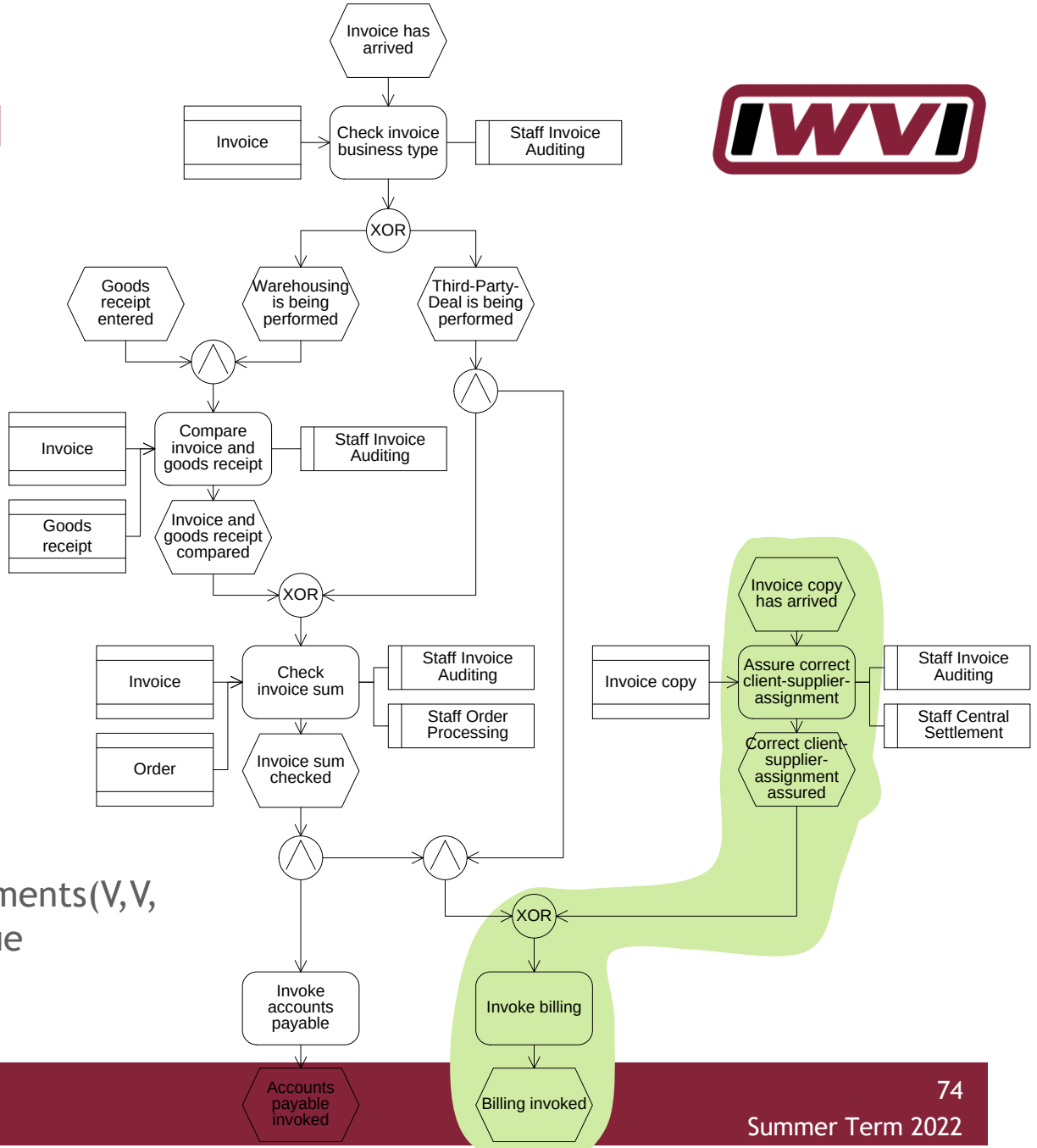
EXAMPLES

- Result: A set of sets!

- Intersection(

  DirectedPaths(
  ElementsOfType(V,event),
  ElementsOfType(V,event)),

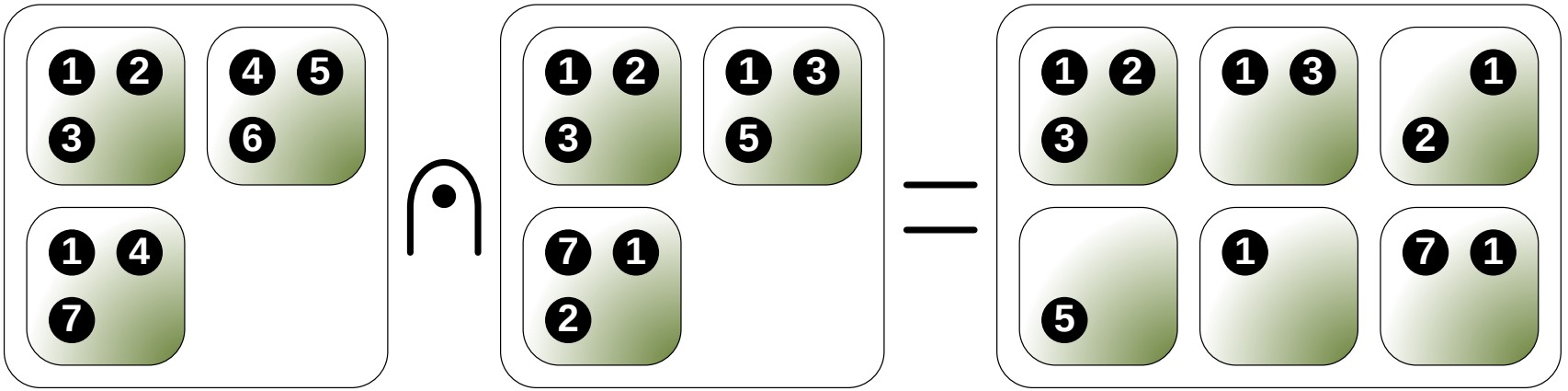
  DirectedPathsContainingElements(V,V,
  ElementsWithArributeOfValue
  (V,label,"*supplier*"))



COMBINATION OF FUNCTION RESULTS

SET OPERATORS

- The INNERINTERSECTION operator performs an intersection on each combination of two inner sets
- If one of the two arguments is a single set, it is handled as a set of sets with each inner set containing one element only (see example on the next slides)



THE INNERINTERSECTION OPERATOR

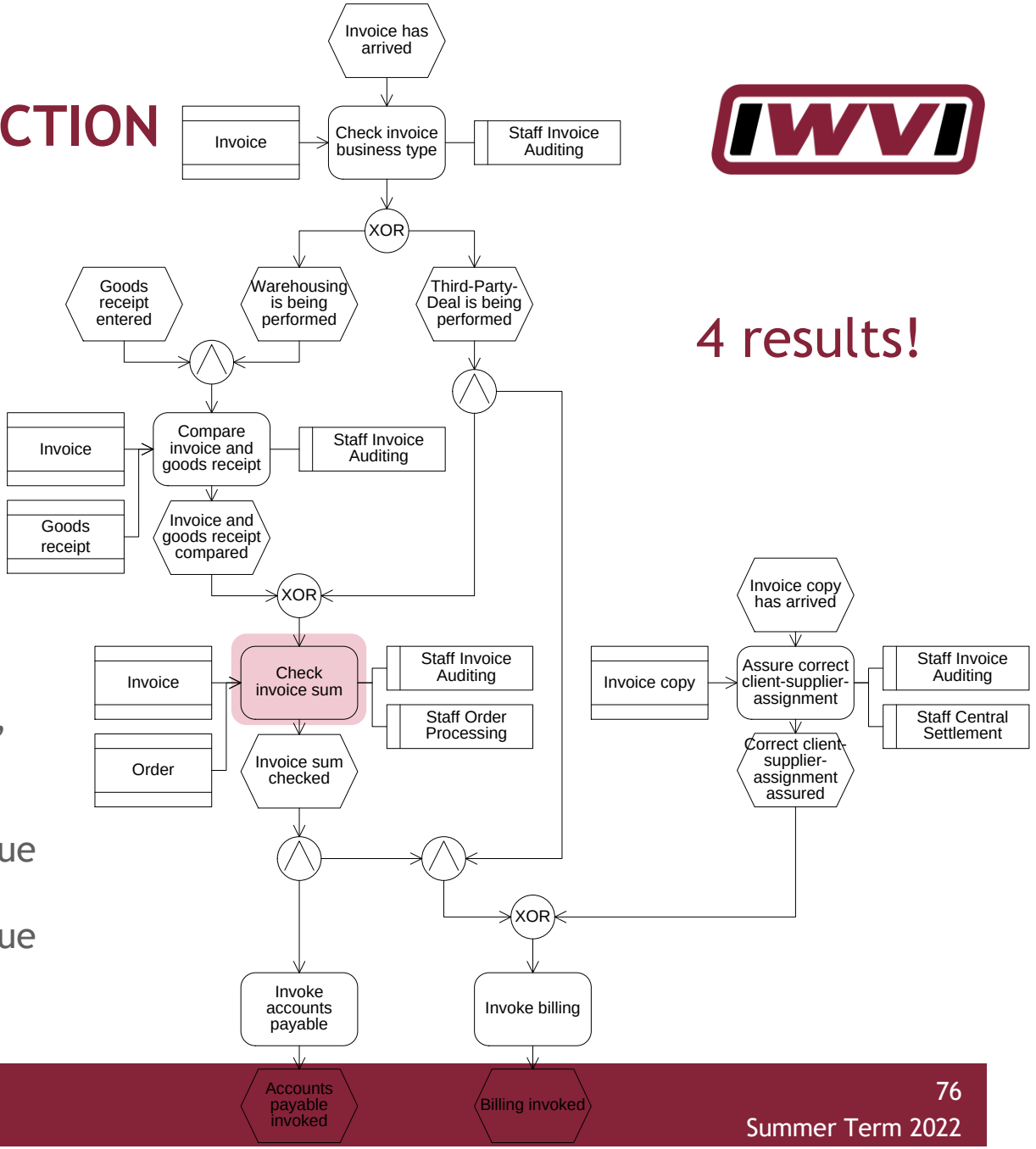
EXAMPLES

- Result: A set of sets!

- InnerIntersection(

  Paths(
  ElementsOfType(V,OrgUnit),
  ElementsOfType(V,OrgUnit)),

  DirectedPaths(
  ElementsWithAttributeOfValue
  (V,label,"Invoice has*"),
  ElementsWithAttributeOfValue
  (V,label, "*invoked")))



4 results!

THE INNERINTERSECTION OPERATOR

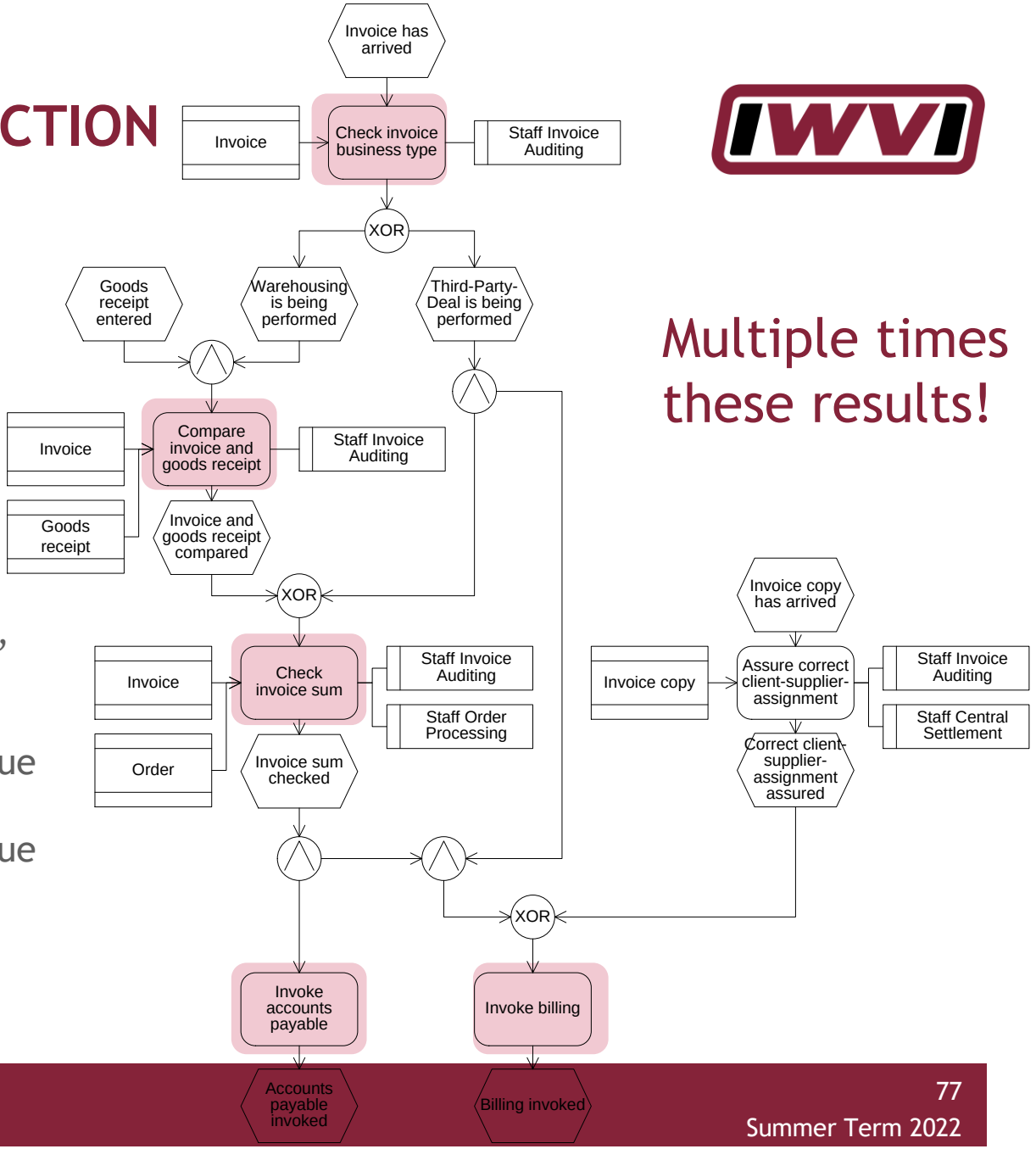
EXAMPLES

- Result: Also a set of sets!

- InnerIntersection(

  ElementsOfType(V,Function),

  DirectedPaths(
  ElementsWithAttributeOfValue
  (V,label,”Invoice has*”),
  ElementsWithAttributeOfValue
  (V,label, ”*invoked”)))

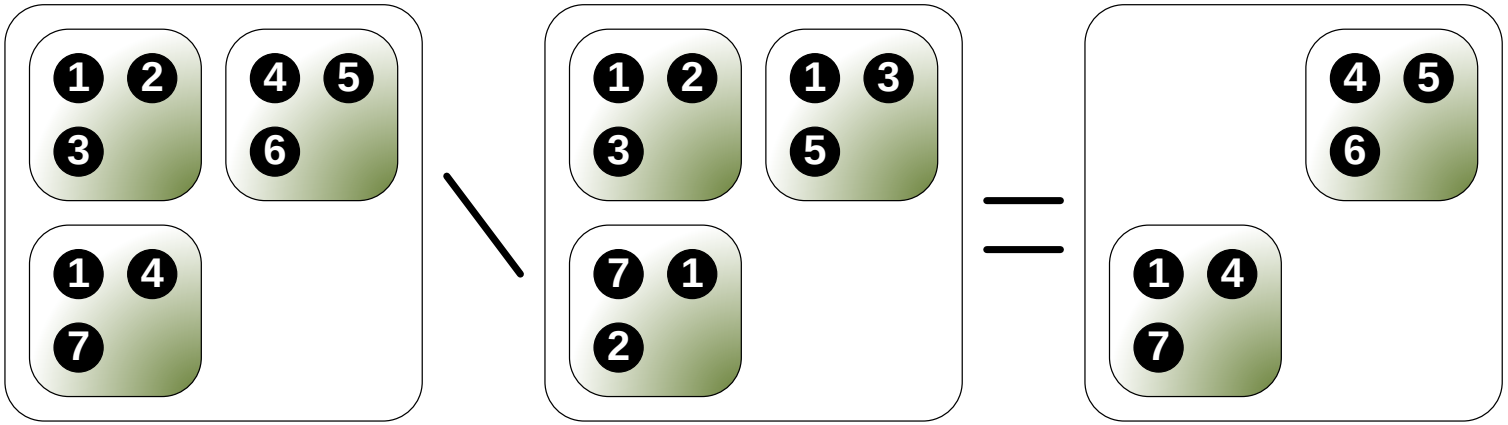


Multiple times these results!

COMBINATION OF FUNCTION RESULTS

SET OPERATORS

- The common COMPLEMENT operator
- Takes either two simple sets or two sets of sets as input

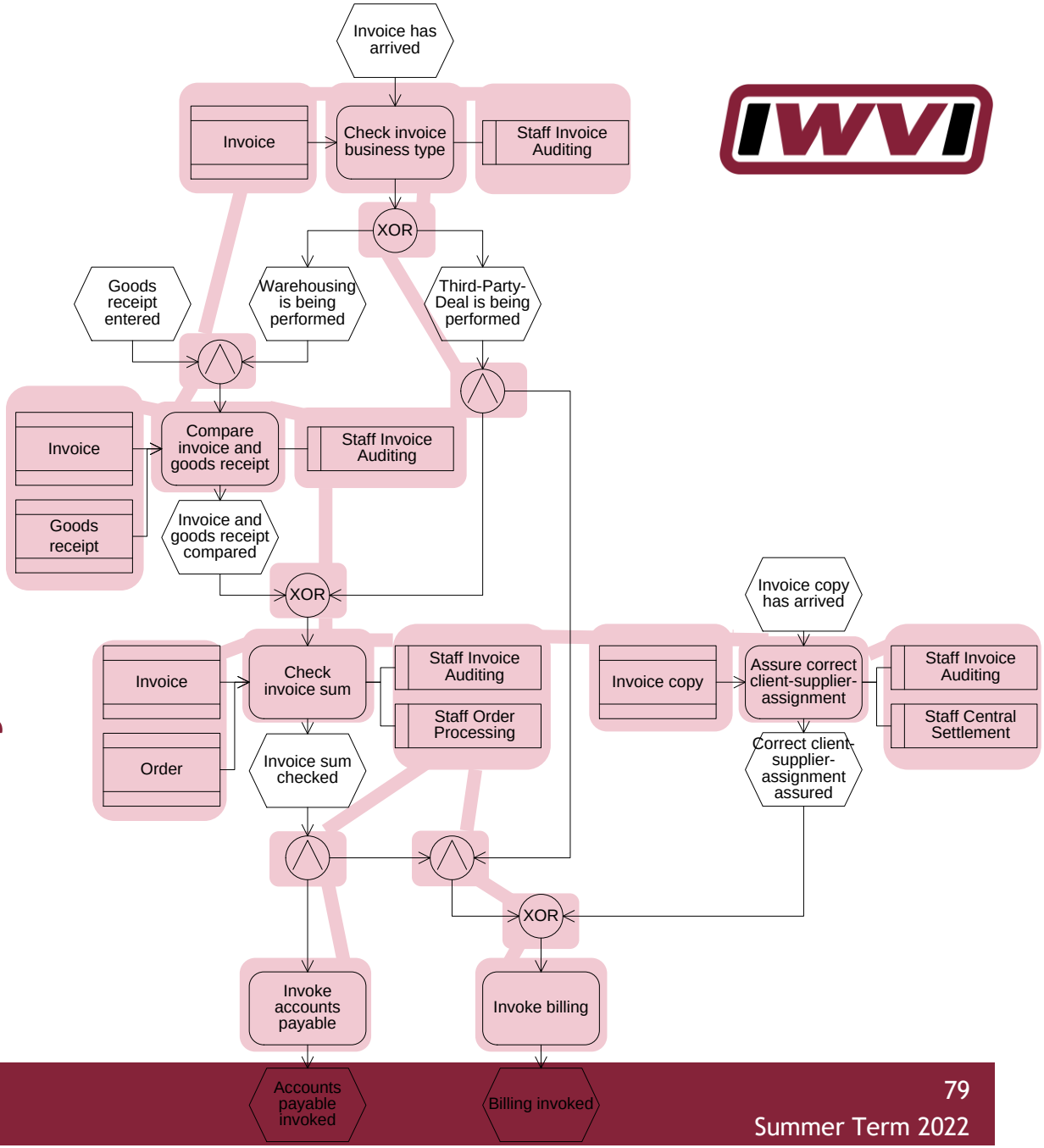


THE COMPLEMENT OPERATOR

EXAMPLES

- Complement(V, ElementsOfType (V,event))

- Result: One simple set! (see single outline in the visulaization)



THE COMPLEMENT OPERATOR

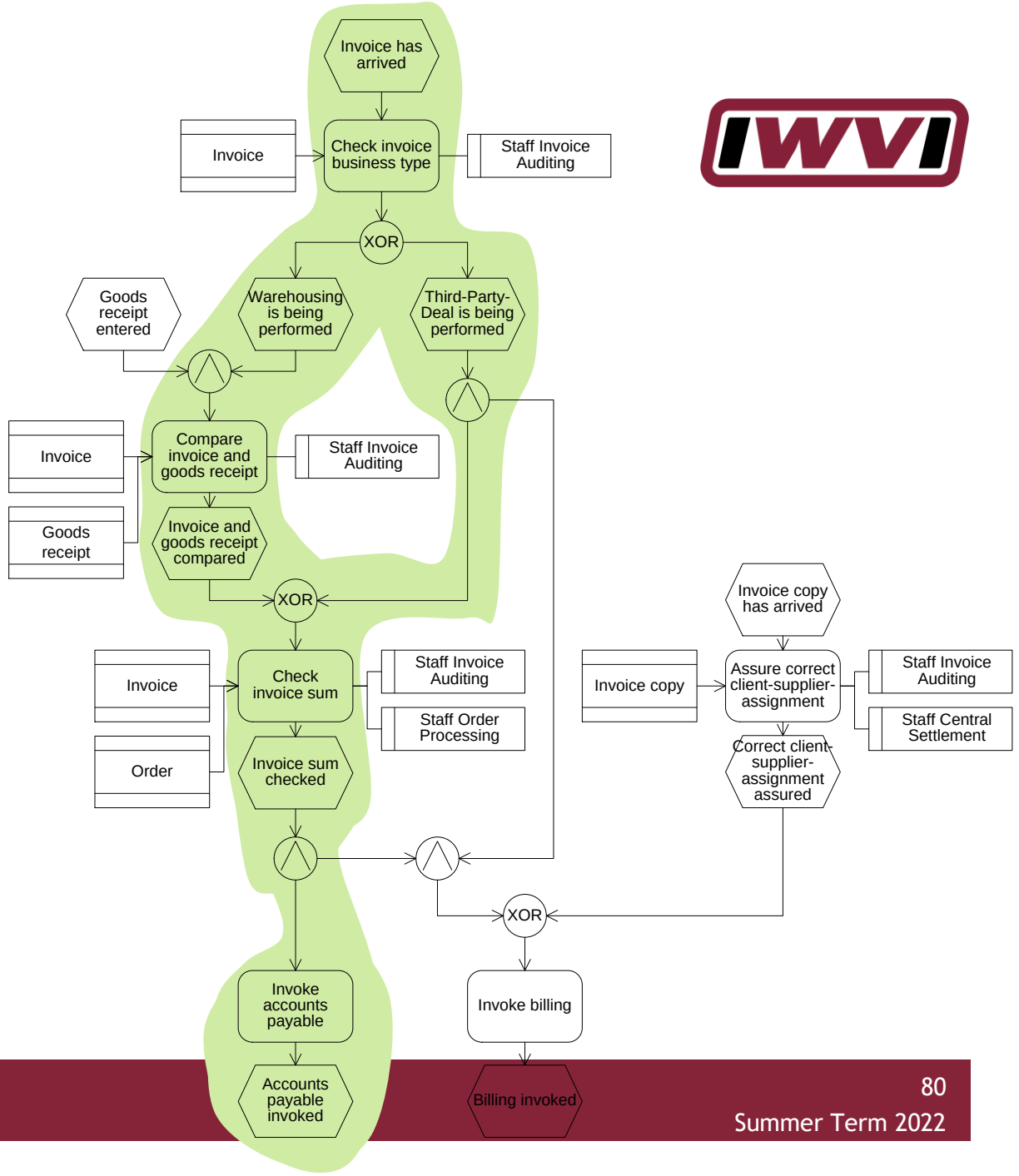
EXAMPLES

- Complement(

  DirectedPaths(Elements-WithAttributeOfValue (V,label,"Invoice has*"), ElementsWithAttributeOf-Value(V,label,"*invoked")),

  DirectedPaths(Elements-WithAttributeOfValue (V,label,"Invoice has*"), ElementsWithAttributeOf-Value(V,label,"Billing*")))

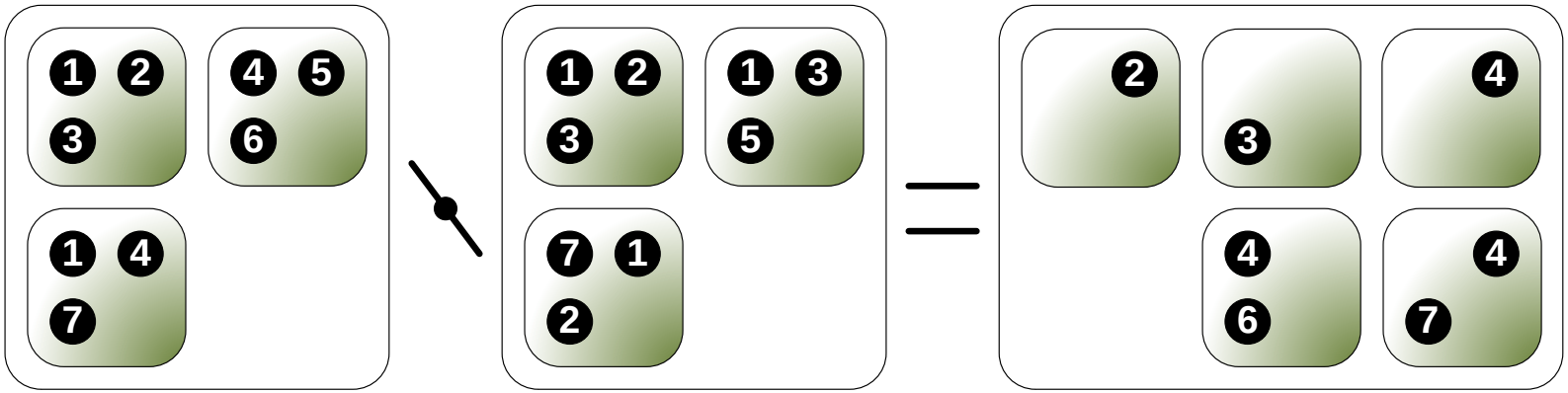
- Result: A set of sets!



COMBINATION OF FUNCTION RESULTS

SET OPERATORS

- The INNERCOMPLEMENT operator performs a subtraction on each inner set pairs having at least one element in common
- If one of the two arguments is a single set, it is handled as a set of sets with each inner set containing one element only



THE INNERCOMPLEMENT OPERATOR

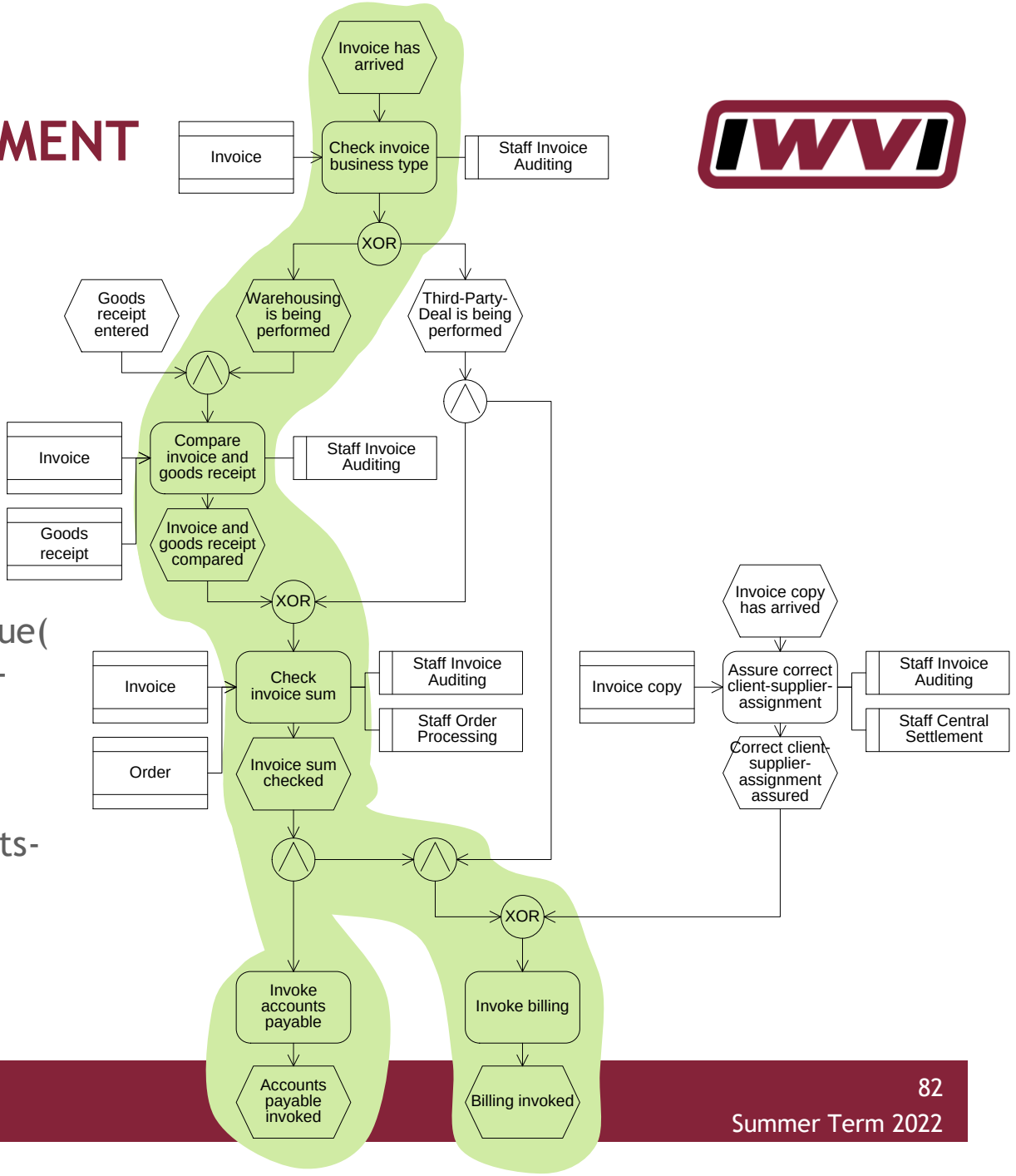
EXAMPLE

- InnerComplement(

  DirectedPathsContaining-Elements(ElementsWith-AttributeOfValue (V,label,"Invoice has*"), ElementsWithAttributeOfValue(V,label,*invoked"),Elements-WithAttributeOfValue (V,label,"Warehousing*"))),

  Adjacent Successors(Elements-WithAttributeOfValue (V,label,"Invoke*"),V)))

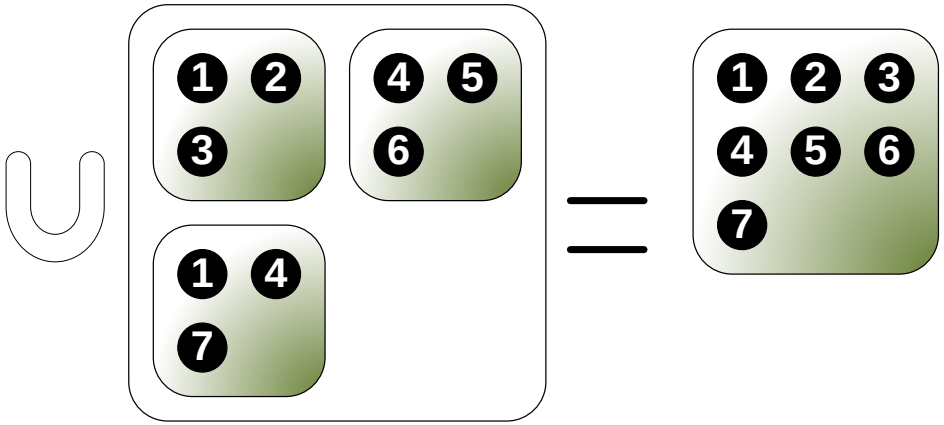
- Result: A set of sets!



COMBINATION OF FUNCTION RESULTS

SET OPERATORS

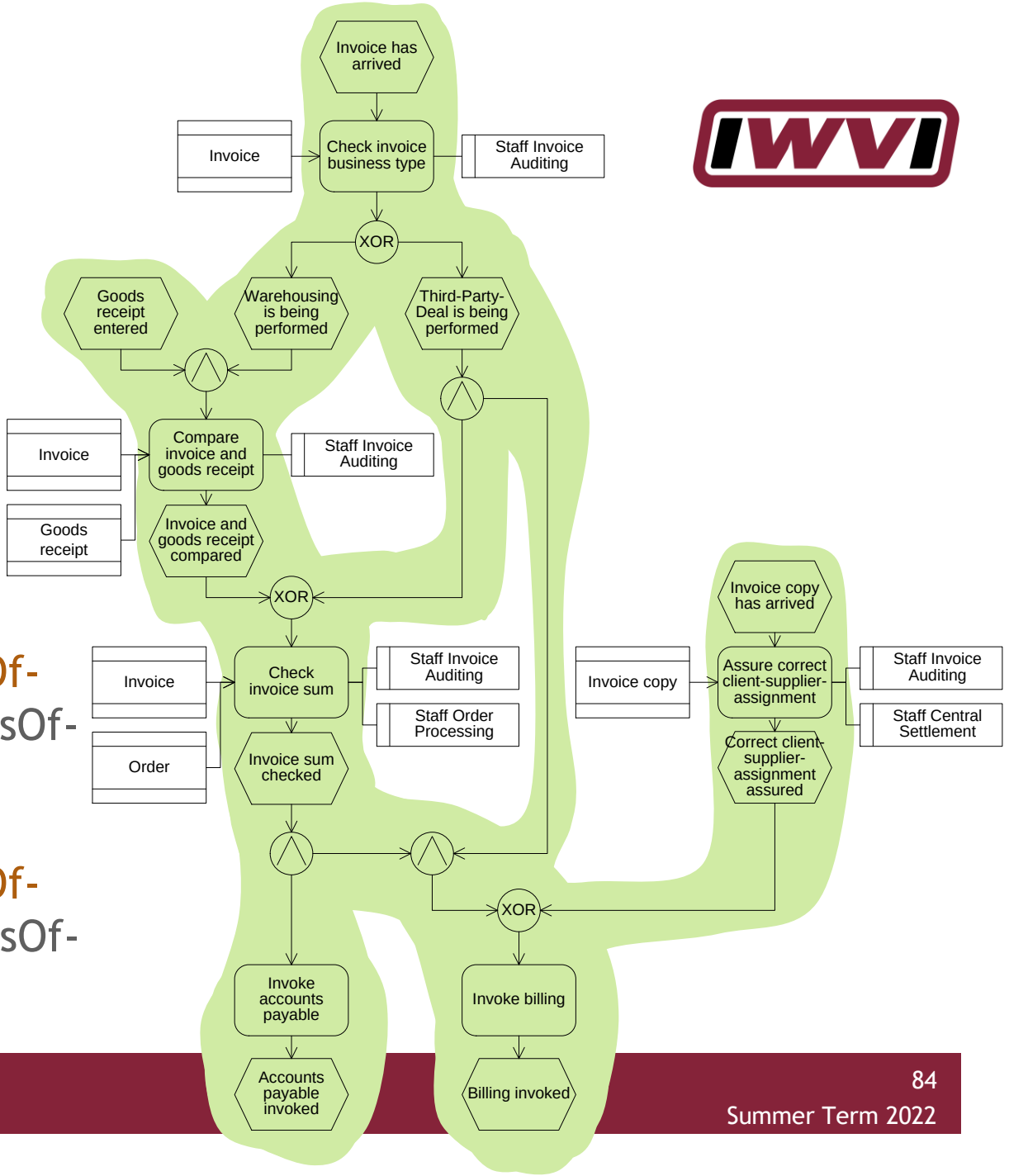
- The SELFUNION operator turns a set of sets into a simple set while performing a union



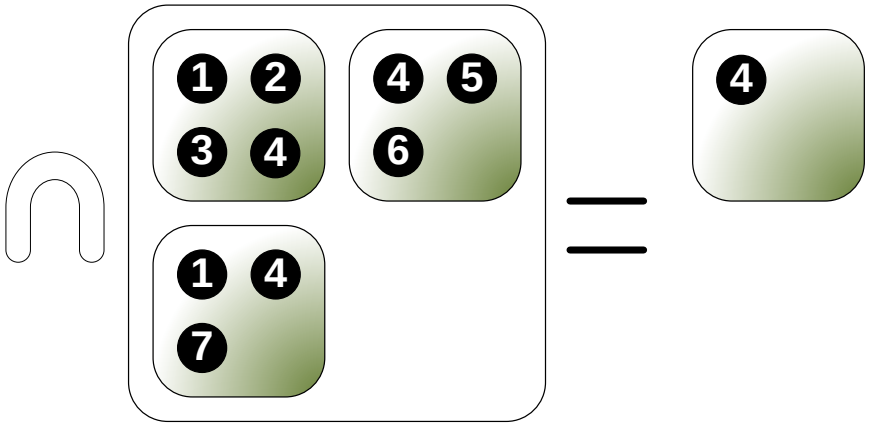
THE SELFUNION OPERATOR

EXAMPLE

- DirectedPaths(SelfUnion(ElementsWithNumberOf-PredRelations(ElementsOf-Type(V,event),0)), SelfUnion(ElementsWithNumberOf-SuccRelations(ElementsOf-Type(V,event),0)))



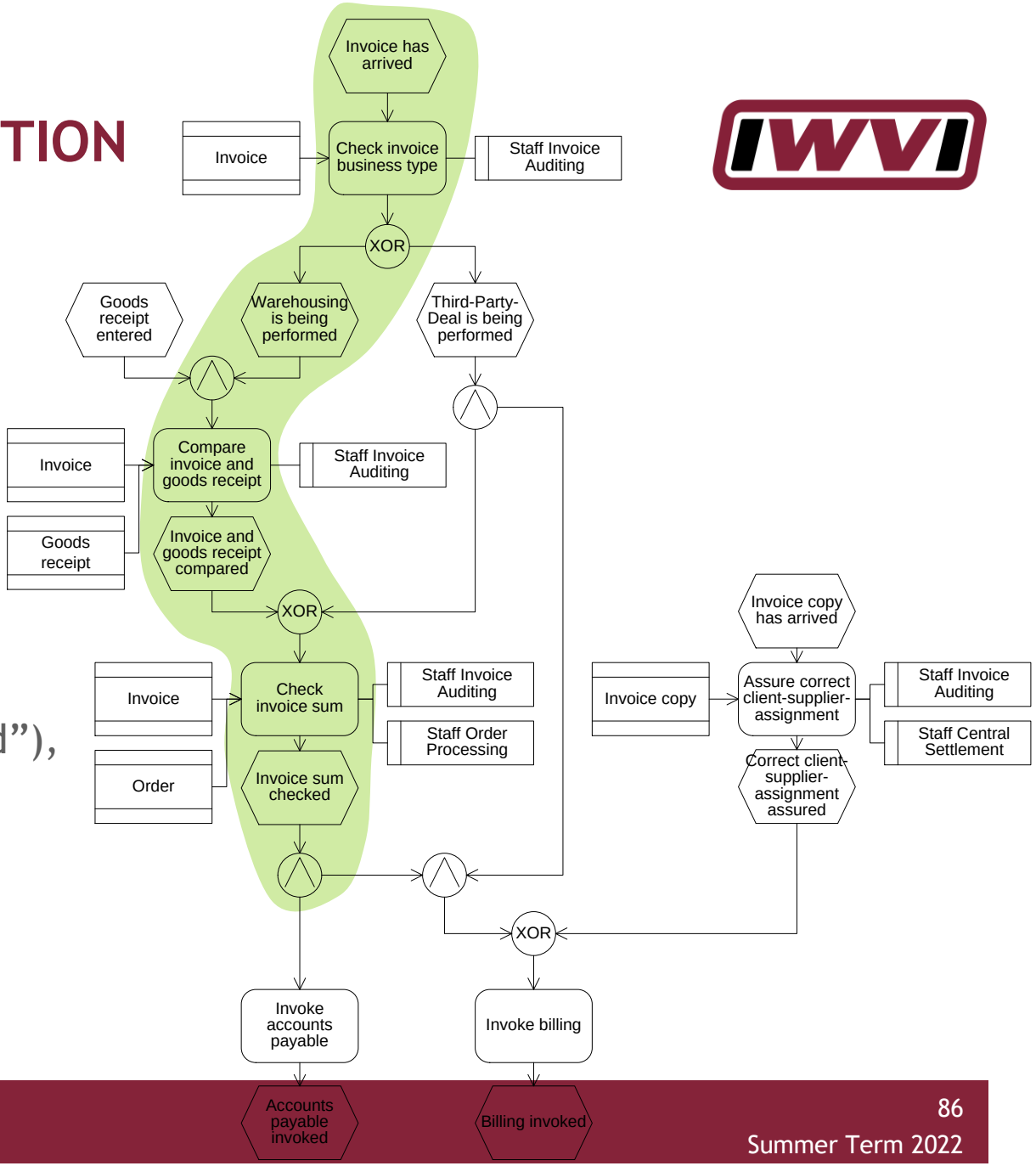
- The SELFINTERSECTION operator turns a set of sets into a simple set while performing an intersection



THE SELFINTERSECTION OPERATOR

EXAMPLE

- SelfIntersection(DirectedPaths-ContainingElements(ElementsWith-AttributeOfValue (V,label,”Invoice has*”), ElementsWithAttribute-OfValue(V,label,”*invoked”), ElementsWithAttribute-OfValue(V,label, ”Warehousing*”))))



GMQL SYNTAX

```
queryExpression = subQueryExpression
    {“,” equationExpression};

subQueryExpression = (functionExpression |
  operatorExpression | setExpression);
```

GMQL SYNTAX

```
functionExpression = functionIdentifier "("
                     subQueryExpression
                  [","  (parameterExpression |
                     subQueryExpression)]
                  [","  (parameterExpression |
                    subQueryExpression)]")";
```

GMQL SYNTAX

```
functionIdentifier = (“ElementsOfType” | “ElementsWithAttributeOfValue”
    | “ElementsWithAttributeOfDatatype” | “ElementsWithRelations” |
      “ElementsWithSuccRelations” | “ElementsWithPredRelations”
 |“ElementsWithRelationsOfType” | “ElementsWithSuccRelationsOfType” |
 “ElementsWithPredRelationsOfType” | “ElementsWithNumberOfRelations” |
                “ElementsWithNumberOfSuccRelations” |
                “ElementsWithNumberOfPredRelations” |
              “ElementsWithNumberOfRelationsOfType” |
            “ElementsWithNumberOfSuccRelationsOfType” |
“ElementsWithNumberOfPredRelationsOfType” | “ElementsDirecltyRelated” |
     “AdjacentSuccessors” | “Paths” | “DirectedPaths” | “Loops” |
         “DirectedLoops” | “ShortestPaths” | “LongestPaths” |
          “ShortestDirectedPaths” | “LongestDirectedPaths” |
    “PathsContainingElements” | “DirectedPathsContainingElements” |
 “PathsNotContainingElements” | “DirectedPathsNotContainingElements” |
    “LoopsContainingElements” | “DirectedLoopsContainingElements” |
 “LoopsNotContainingElements” | “DirectedLoopsNotContainingElements”);
```

GMQL SYNTAX

```
parameterExpression = (Integer | ElementType
| AttributeDataType |
AttributeValue | Variable);

ElementType = String;

AttributeDataType = (“INTEGER” | “STRING” |
“BOOLEAN” | “ENUM” | “DOUBLE”);

AttributeValue = String;
```

GMQL SYNTAX

```
operatorExpression = operatorIdentifier "(" 
                     subQueryExpression
                     ["," subQueryExpression] ")";

operatorIdentifier = ("UNION" | 
"INTERSECTION" | "COMPLEMENT" | "JOIN" | 
"INNERINTERSECTION" | "INNERCOMPLEMENT" | 
"SELFUNION" | "SELFINTERSECTION");

setExpression = ("V" | "E" | "Z");
```

GMQL SYNTAX

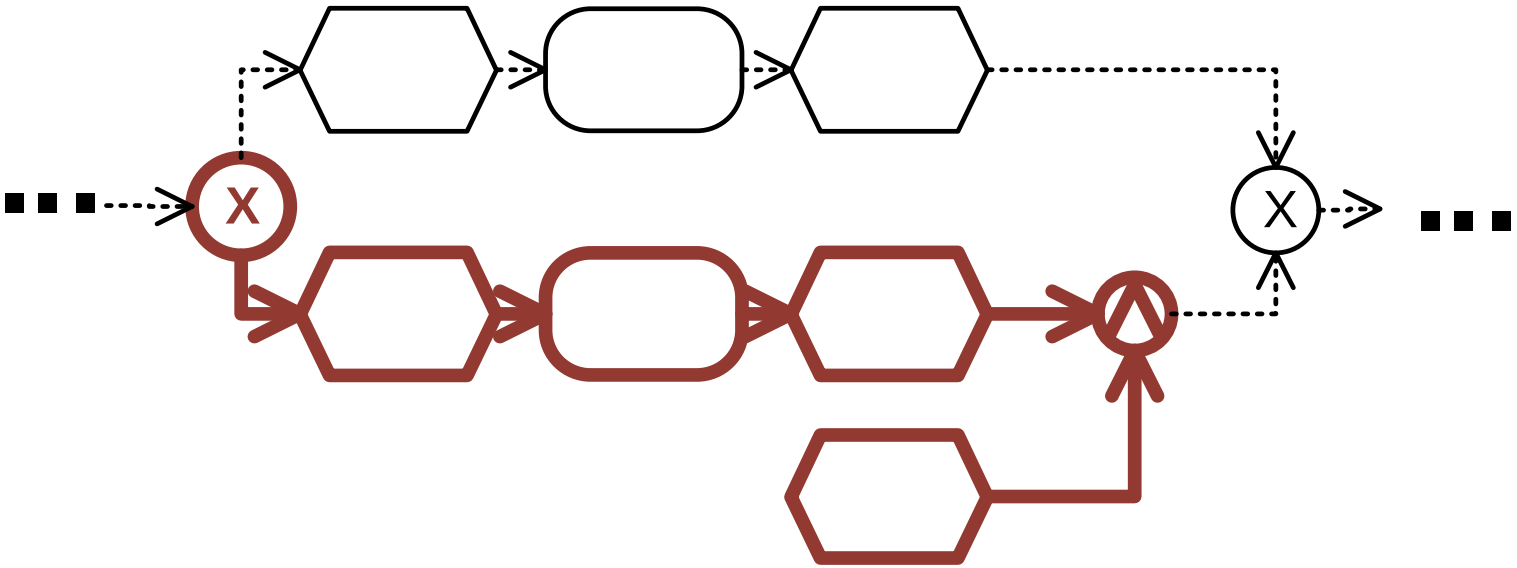
```
Variable = "("("A" | "B" | "C" | "D" | "E" |
  "F" | "G" | "H" | "I" | "J" | "K" | "L" |
  "M" | "N" | "O" | "P" | "Q" | "R" | "S" |
  "T" | "U" | "V" | "W" | "X" | "Y" | "Z")+
                "," ("Integer" |
    "ElementType" | "AttributeDataType" |
            "AttributeValue")")";

equationExpression = Variable ("=" | "!=" |
     "<" |  ">" | "<=" | ">=") Variable;
```

AGENDA

- Requirements of Model Query Languages
- The Generic Model Query Language (GMQL)
- Example Queries
- Live Demo

EXAMPLE QUERIES

AND MIGHT NOT GET CONTROL FROM XOR (CF. MENDLING 2007)



EXAMPLE QUERIES

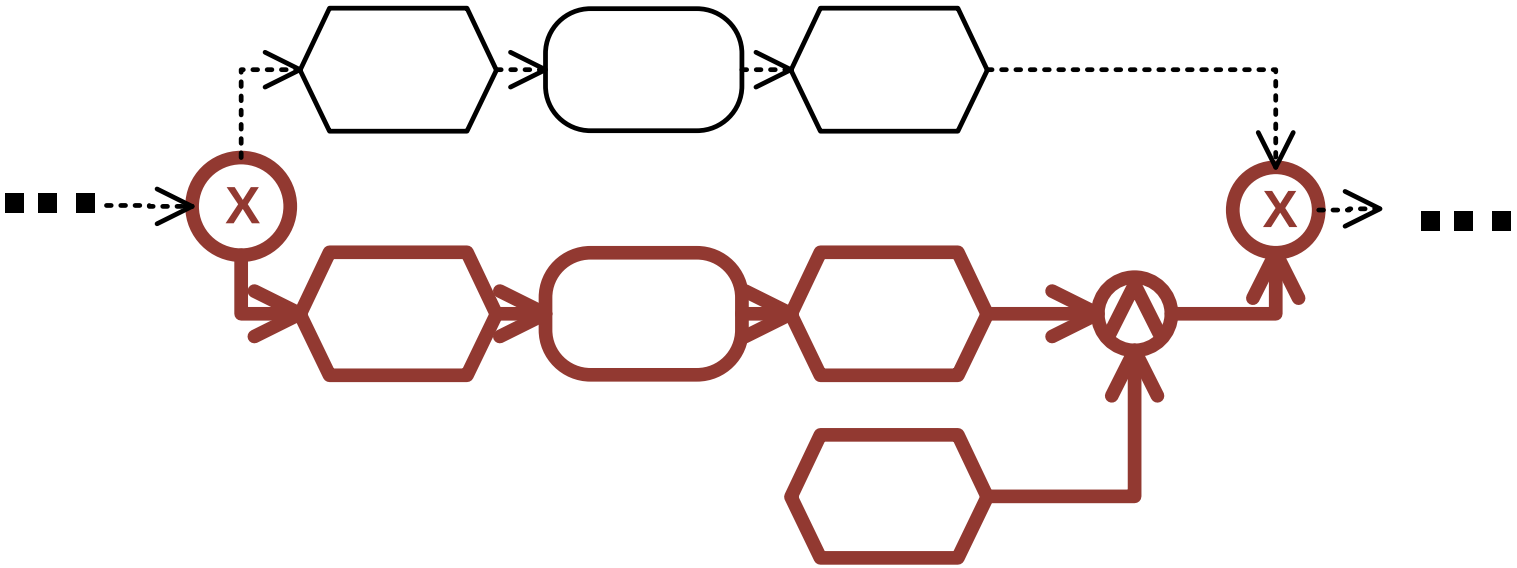
AND MIGHT NOT GET CONTROL FROM XOR (CF. MENDLING 2007)

```
DirectedPaths(
    COMPLEMENT(
        ElementsOfType(V,XOR),
        UNION(
            SELFUNION(INNERINTERSECTION(
                ElementsOfType(V,XOR),
                ElementsWithNumberOfSuccRelations(ElementsOfType(V,XOR),1))),
            SELFUNION(ElementsWithNumberOfSuccRelations(ElementsOfType(V,XOR),0))
        )
    ),
    SELFUNION(INNERINTERSECTION(
        ElementsOfType(V,AND),
        AdjacentSuccessors(
            SELFUNION(ElementsWithNumberOfPredRelations(ElementsOfType(V,Event),0)),
            ElementsOfType(V,AND)
        )
    ))
)
```

EXAMPLE QUERIES

AND MIGHT NOT GET CONTROL FROM XOR (CF. MENDLING 2007)

- "Extended version" of the pattern



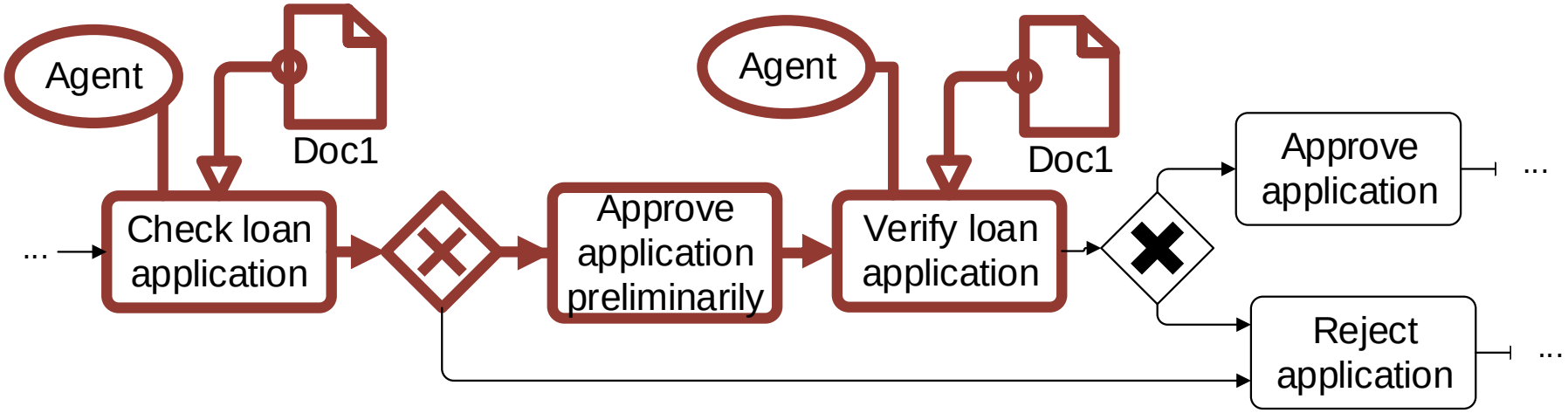
EXAMPLE QUERIES

AND MIGHT NOT GET CONTROL FROM XOR (CF. MENDLING 2007)

```
DirectedPathsContainingElements(
        COMPLEMENT(
                ElementsOfType(V,XOR),
                UNION(
                        SELFUNION(INNERINTERSECTION(
                                ElementsOfType(V,XOR),
                                ElementsWithNumberOfSuccRelations(ElementsOfType(V,XOR),1))),
                        SELFUNION(ElementsWithNumberOfSuccRelations(ElementsOfType(V,XOR),0))
                )
        ),
        COMPLEMENT(
                ElementsOfType(V,XOR),
                UNION(
                        SELFUNION(INNERINTERSECTION(
                                ElementsOfType(V,XOR),
                                ElementsWithNumberOfPredRelations(ElementsOfType(V,XOR),1))),
                        SELFUNION(ElementsWithNumberOfPredRelations(ElementsOfType(V,XOR),0))
                )
        ),
        SELFUNION(INNERINTERSECTION(
                ElementsOfType(V,AND),
                AdjacentSuccessors(
                        SELFUNION(ElementsWithNumberOfPredRelations(ElementsOfType(V,Event),0)),
                        ElementsOfType(V,AND)
                )
        ))
)
```

EXAMPLE QUERIES

SEPARATION OF DUTIES (CF. KNORR AND STORMER 2001)



EXAMPLE QUERIES

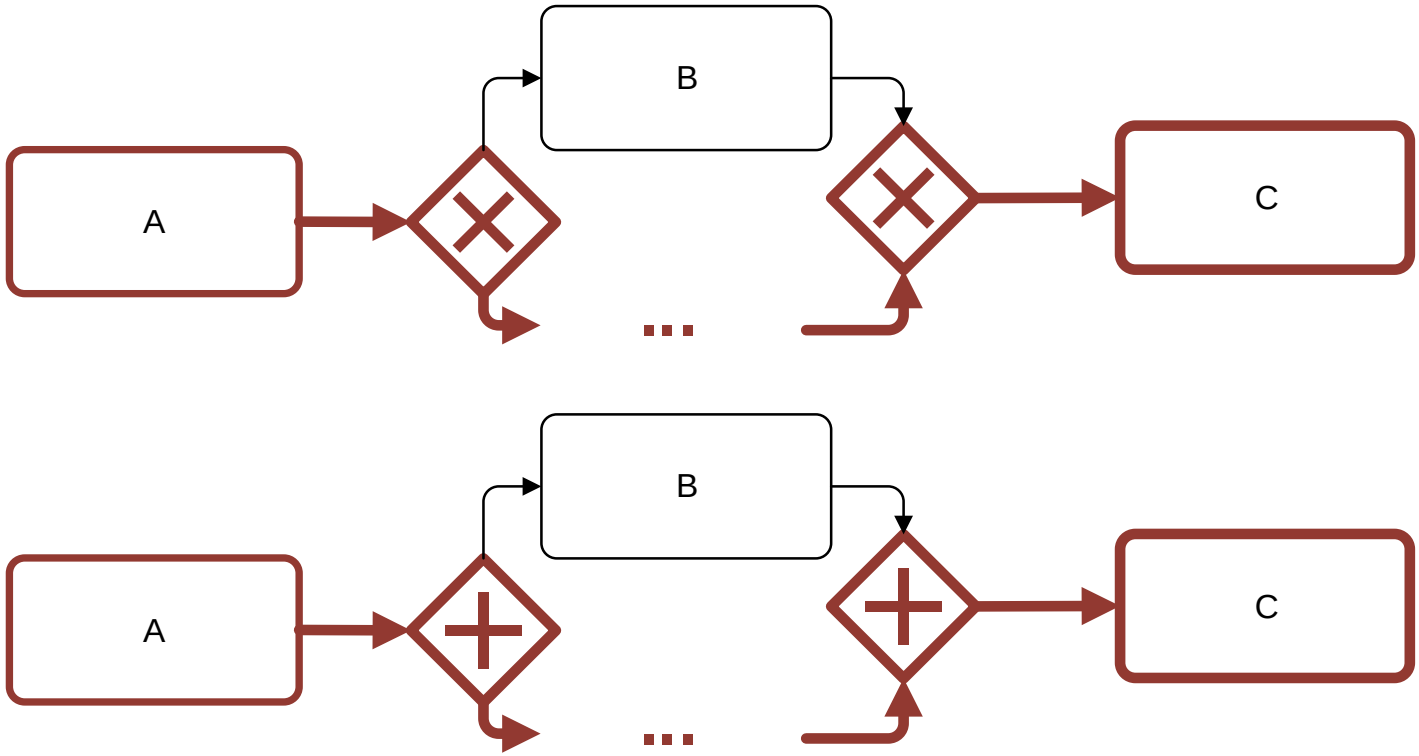
SEPARATION OF DUTIES (CF. KNORR AND STORMER 2001)

```
DirectedPaths(
  SELFUNION(INNERINTERSECTION(
    AdjacentSuccessors(
        ElementsWithAttributeOfValue(ElementsOfType(V, Document), Label, (A,AttributeValue)),
        ElementsOfType(V, Activity)),
    ElementsDirectlyRelated(
        ElementsWithAttributeOfValue(ElementsOfType(V, OrgaUnit), Label, (C,AttributeValue)),
        ElementsOfType(V, Activity)))),
  SELFUNION(INNERINTERSECTION(
    AdjacentSuccessors(
        ElementsWithAttributeOfValue(ElementsOfType(V, Document), Label, (B,AttributeValue)),
        ElementsOfType(V, Activity)),
    ElementsDirectlyRelated(
        ElementsWithAttributeOfValue(ElementsOfType(V, OrgaUnit), Label, (D,AttributeValue)),
        ElementsOfType(V, Activity))))
),
```

(A,AttributeValue)=(B,AttributeValue), (C,AttributeValue)=(D,AttribteValue)

EXAMPLE QUERIES

PREDECESSOR / SUCCESSOR RULE (CF. AWAD ET AL. 2008)



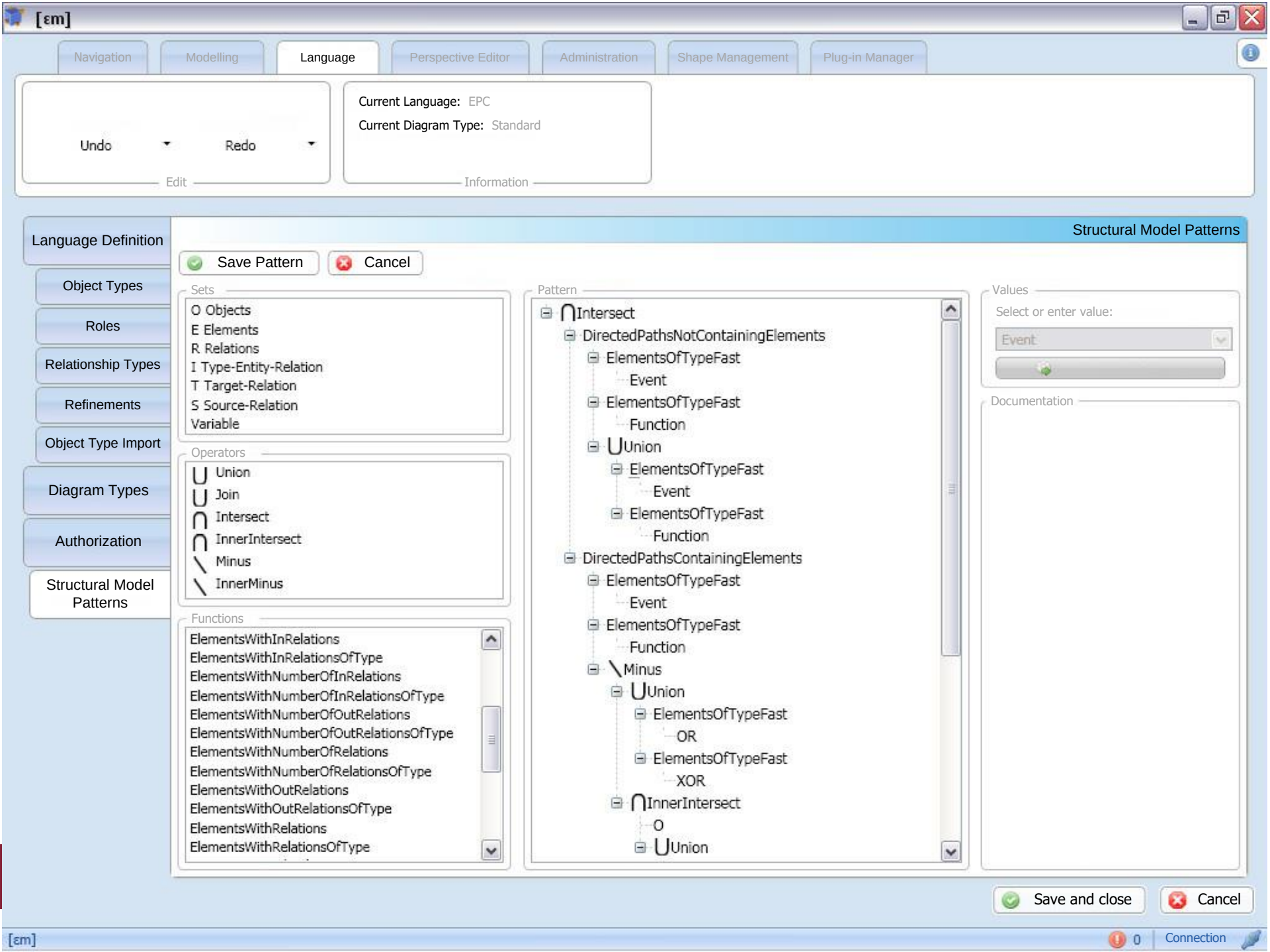
EXAMPLE QUERIES

PREDECESSOR / SUCCESSOR RULE (CF. AWAD ET AL. 2008)

```
COMPLEMENT(
  DirectedPathsNotContainingElements(
      ElementsWithAttributeOfValue(ElementsOfType(V,Activity), Label,„A“),
      ElementsWithAttributeOfValue(ElementsOfType(V,Activity), Label,“C“),
      ElementsWithAttributeOfValue (ElementsOfType(V,Activity), Label,“B“)),
  DirectedPathsContainingElements(
      ElementsWithAttributeOfValue(ElementsOfType(V,Activity), Label,“A“),
      ElementsWithAttributeOfValue (ElementsOfType(V,Activity), Label,“C“),
      INNERINTERSECTION(
          AdjacentSuccessors(
              ElementsOfType(V,AND),
              ElementsWithAttributeOfValue(ElementsOfType(V,Activity), Label,“B“)),
          ElementsOfType(V,AND))
))
```

AGENDA

- Requirements of Model Query Languages
- The Generic Model Query Language (GMQL)
- Example Queries
- Live Demo


[εm]
Navigation
Modelling
Language
Perspective Editor
Administration
Shape Management
Plug-in Manager
Undo
Redo
Edit
Current Language: EPC
Current Diagram Type: Standard
Information
Language Definition
Object Types
Roles
Relationship Types
Refinements
Object Type Import
Diagram Types
Authorization
Structural Model Patterns
Structural Model Patterns
Save Pattern
Cancel
Sets
O Objects
E Elements
R Relations
I Type-Entity-Relation
T Target-Relation
S Source-Relation
Variable
Operators
Union
Join
Intersect
InnerIntersect
Minus
InnerMinus
Functions
ElementsWithInRelations
ElementsWithInRelationsOfType
ElementsWithNumberOfInRelations
ElementsWithNumberOfInRelationsOfType
ElementsWithNumberOfOutRelations
ElementsWithNumberOfOutRelationsOfType
ElementsWithNumberOfRelations
ElementsWithNumberOfRelationsOfType
ElementsWithOutRelations
ElementsWithOutRelationsOfType
ElementsWithRelations
ElementsWithRelationsOfType
Pattern
Intersect
DirectedPathsNotContainingElements
ElementsOfTypeFast
Event
ElementsOfTypeFast
Function
Union
ElementsOfTypeFast
Event
ElementsOfTypeFast
Function
DirectedPathsContainingElements
ElementsOfTypeFast
Event
ElementsOfTypeFast
Function
Minus
Union
ElementsOfTypeFast
OR
ElementsOfTypeFast
XOR
InnerIntersect
O
Union
Values
Select or enter value:
Event
Documentation
Save and close
Cancel
[εm]
0
Connection

[ɛm]

Navigation | Modeling | Language Editor | Perspective Editor | Administration | Shape Management | Plug-in Manager

Edit
Save
Close Model
Model

Select | Connect
Undo | Redo
Modeling

Zoom in | Zoom out
Width | Page
75%
View

Show shapes | Print | Grid
Layout | Connection points | Page setup
Export
Modeling Environment

Pattern: Decision split after event in path of arbitrary length
Model selection: Invoice Auditing (Standard)
Matches: 1 / 2
New search in: Model Selection
Pattern Matching

Invoice Auditing

Invoice has arrived
Determine invoice business type
Staff invoice auditing
Goods receipt entered
Business type is warehousing
Business type is 3rd-party-deal
Compare invoice and goods receipt
Staff invoice auditing
Invoice and goods receipt compared
Invoice copy has arrived
Check invoice sum
Staff invoice auditing
Staff order processing
Check assignment of client and supplier
Staff invoice auditing
Staff central settlement
Invoice sum checked
Assignment of client and supplier checked
Transmit invoice to accounts payable
Staff invoice auditing
Transmit invoice to billing
Staff invoice auditing
Invoice transmitted to accounts payable
Invoice transmitted to billing

[em] Structural Pattern Matching

[ɛm]
Pattern selection

Suche:

Complete paths — Sprache: EPC
Decision split after event in path of arbitrary length — Sprache: EPC
Connector loops — Sprache: EPC
Join-and-split connectors — Sprache: EPC

Search | Cancel

[ɛm]
0 | Connection

REFERENCES

- GMQL
  Delfmann, P.; Steinhorst, M.; Dietrich, H.-A.; Becker, J.:
  The generic model query language GMQL – Conceptual specification, implementation, and runtime evaluation.
  Information Systems 47 (2015) 1, pp. 129-177.

- Related Model Query Languages
  See related work section in the above article!

UNIVERSITÄT KOBLENZ · LANDAU

IWVI

BUSINESS PROCESS MANAGEMENT

MODEL QUERY II:
THE GENERIC MODEL QUERY LANGUAGE (GRAPH MATCHING)

INSTITUTE FOR IS RESEARCH

www.uni-koblenz.de