

# Exploratory Didactics by Plug and Play in Prolog

**Klaus P. Jantke**

ADICOM Software

Weimar, Germany

*[Klaus.P.Jantke@adicom-group.de](mailto:Klaus.P.Jantke@adicom-group.de)*

**Oksana Arnold**

Erfurt University of Applied  
Sciences

Erfurt, Germany

*[oksana.arnold@fh-erfurt.d](mailto:oksana.arnold@fh-erfurt.d)*

**Jürgen Cleve**

Wismar University of Applied  
Sciences

Wismar, Germany

*[juergen.cleve@hs-erfurt.d](mailto:juergen.cleve@hs-erfurt.d)*

**Rainer Knauf**

Ilmenau University of  
Technology

Ilmenau, Germany

*[rainer.knauf@tu-ilmenau.de](mailto:rainer.knauf@tu-ilmenau.de)*

# 1. Educational Gamification

applications:

- staff training for civil protection and crisis management
- employees training for the prevention of accidents in industry
- ...

the introduction of game-based learning and training is a process of **gamification by transformation**

## Gamification

- based on
  - learning material such as
    - media documents
    - opening of a tool (URL, an e-learning system, ...)
    - informal activity description
  - proven didactic principles
- aims at playful learning and training experiences that are attractive (like digital games) to learners
- stimulate human communication about interesting experiences (like digital games), and that are highly effective (like digital games).

Educational gamification includes the design of learning and game play experiences. But it is not design out of nothing. People are already teaching, coaching, learning, training ... Gamification means to take what is already there and to transform it such that more affective and more effective experiences become possible and likely.

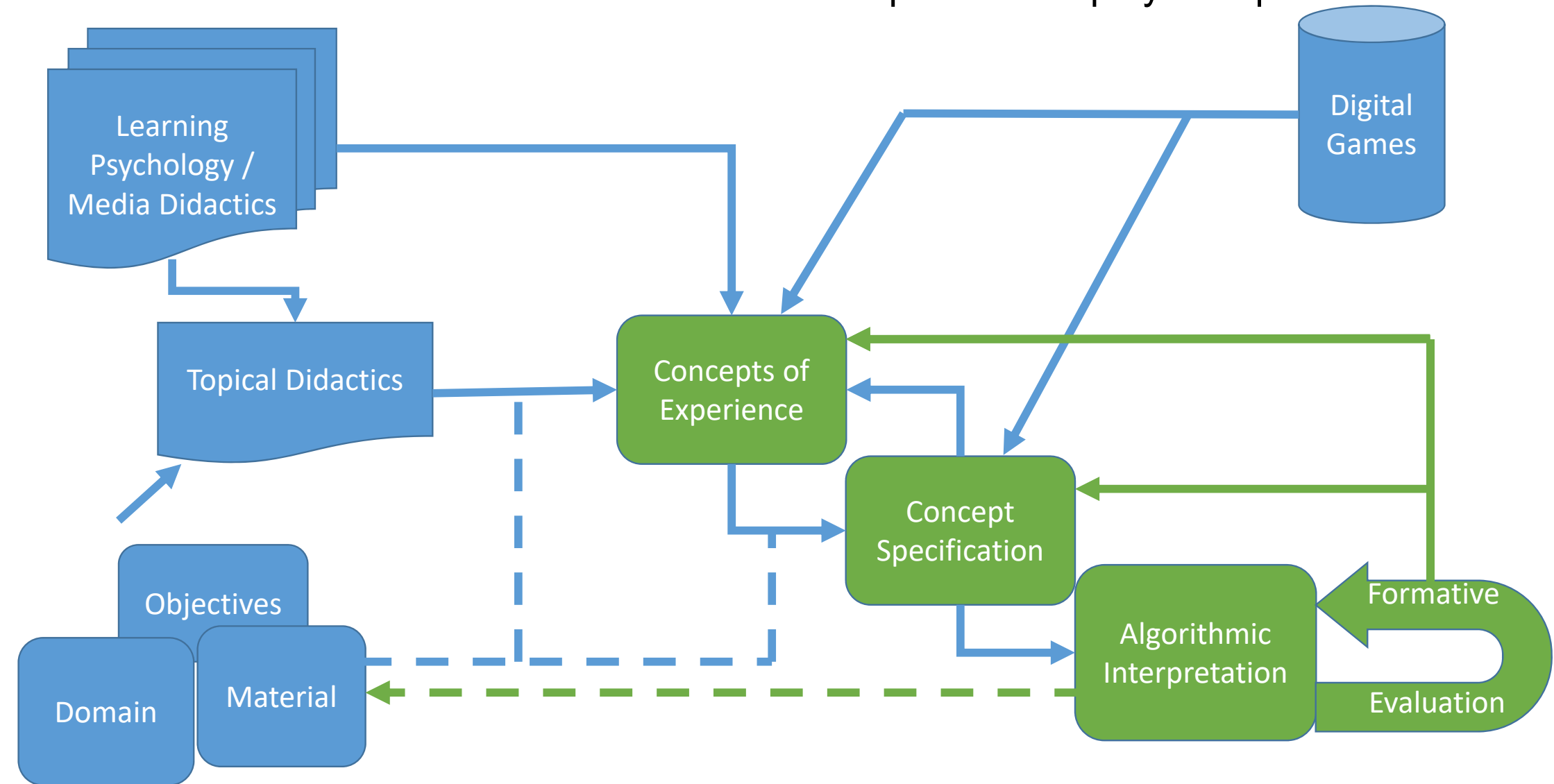
Among the prerequisites of the transformation, there is

- experience with teaching and training so far
- material such as text, video, pictures, up to implementations including simulations
- general didactics and topical didactics
- experiences of game play.

The confluence of didactics and ludology is a novelty. The idea is revolutionary, because conventional pedagogy sees itself as something serious as opposed to anything playful.

To sum up, it is the **“art” of designing affective and effective experiences**

Educational Gamification as transformation of learning / training material and/or educational environment into a form that bears the potential of playful experiences



## 2 Concept specification

### 2.1 Modeling Didactics by Storyboarding

What is **DIDACTICS** ? = *(meta) knowledge about proper knowledge transmission*

- Completely informal (so far)
- Much of it is not represented at all (*just utilized by experienced teachers*)

- *Let's make explicit what we talk about!*
  - ⇒ (semi-) formal **represent** didactics
- *Let's apply such representations in (our university) practice!*
  - ⇒ enable also non-experts in didactics to **process** a model of didactics
- *Let's explore conditions that can be (formally) checked: consistency conditions, invariants, didactical principles, ... !*
  - ⇒ **verify** didactic knowledge
- *Let's check the result of applying certain didactics in a case study!*
  - ⇒ **validate** applied didactics based on the degree of success
- *Let's learn from the validation results!*
  - ⇒ **refine** didactic knowledge towards incremental improvement
- *Let's derive successful didactic patterns!*
  - ⇒ **learn** didactic patterns inductively from successful and failing examples
- *Let's utilize these patterns!*
  - ⇒ **support** didactics by a design tool with a pattern library

**our approach towards doing the above**

**Storyboarding** - a modeling concept for Didactic Knowledge




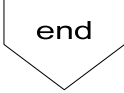
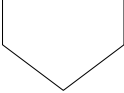
## Storyboard Elements

- **Scenes** non-decomposable learning activity implemented in any way, e.g.
  - presentation of a (media) document,
  - opening of a tool (URL, an e-learning system, ...)
  - informal activity description
- **Episodes** defined by their sub-graph
- **Graphs** interpreted by the paths, on which they can be traversed
- **Start Node** defines starting point of legal graph traversing
- **End Node** defines target point of a legal graph traversing
- **Edges** transitions between nodes, rules:
  - (1) The outgoing edge must have the same color as the incoming edge by which the node was reached.
  - (2) If there is a condition specified as the edge's key attribute, this condition has to be met for leaving the node by this edge.
- **Key attributes of nodes** specify application driven information for all nodes of the same type
- **Key attributes of edges** specify conditions, which have to be true for continuing traversing on this edge
- **Free attributes** specify whatever the storyboard author wants the user to know: didactic intentions, useful methods, necessary equipment, ...

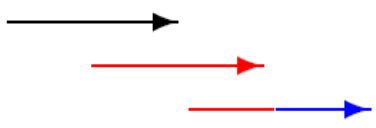
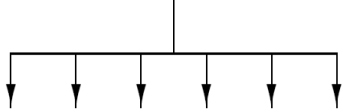
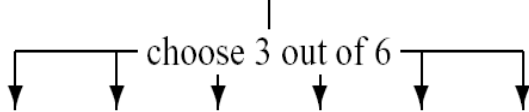
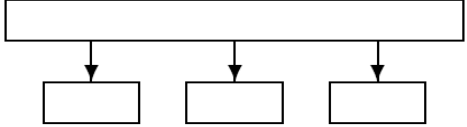
## This way, storyboarding provides

- **Clarity** in how the agent model behaves via high-level modeling
- **Simplicity** of use for the agent developer and validator.
- Intuitive **visual appearance** for ease of understanding of agent behaviors

## Node types

	Scene	Episode	Start Node	End Node	Reference Node
Symbol					
Behavior on double click	<ul style="list-style-type: none"> <li>opening a document</li> <li>nothing, if just verbal activity description</li> </ul>	opening the related sub-graph	jump to the <b>Start Node</b> of the related super-graph	jump to the <b>Reference Node</b> that succeeds it's associated <b>Episode Node</b> in the related super-graph	jump to the <b>End Node</b> of the sub-graph that is associated to the preceded <b>Episode Node</b>
Behavior on following a hyperlink	<ul style="list-style-type: none"> <li>opening a document</li> <li>visiting a website, if URL</li> <li>opening the mail tool, if email address</li> </ul>	<ul style="list-style-type: none"> <li>opening a document</li> <li>visiting a website, if URL</li> <li>opening the mail tool, if email address</li> </ul>	not meaningful		

## Edge types

	Simple Edge	Fork	Fork with conditions	Alternatives
Symbol				
Inter-pretation	defines a unique successor node	defines several successor nodes, which are traversed independently in any sequence	defines several successor nodes, which are traversed independently in any sequence, but according to the specified condition	defines several successor nodes, out of which exactly one has to be traversed



## KE Method # 1: Formal Verification of Storyboards

### 1. Hierarchy Completeness Test

- *Does every episode have exactly one related graph?*
- *Does every (non-top) graph have exactly one related episode node in exactly one related super-graph?*

### 2. Path Completeness Test

- *Does every traversing path terminate?*

*In other words:*

*Is the **End Node** reachable on every possible path in each (sub-) graph?*

- *Is each node reachable from the **Start Node** in each (sub-) graph?*

### 3. Node Soundness Test

- *Are alternative outgoing edges (of the same beginning color) logically complete and consistent?*

### 4. Interdependence of Incoming/Outgoing Edges (Edge Color Test)

- *Is there a unique start color?*

*In other words:*

*Is there a unique (beginning) color of the **Start Node**'s outgoing edges?*

- *Is there at least one outgoing edge with the same (beginning) color for each incoming edges' (finishing) colors?*

## **KE Method # 2: Annotation Heritage**

- In some applications it makes sense to inherit annotations from nodes (both scenes and episodes) to their related super-graph, e.g.
  - Material that are used to teach a particular lecture is also material to teach the complete subject the lecture is part of
- In other cases it makes sense to inherit the arithmetic sum of a key annotation of all nodes to the related super-graph, e.g.
  - An upper limit of the time needed to teach a subject can be estimated by the sum of its components (lectures)
  - A maximum cost of a university study can be estimated by the sum of the fees for all recommended subjects
- In other cases it makes sense to inherit the maximum value of a key annotation of all nodes to the related super-graph
  - The educational difficulty (basic/easy, medium, advanced, very difficult) of a study needs to be communicated as the maximum value of all mandatory subjects

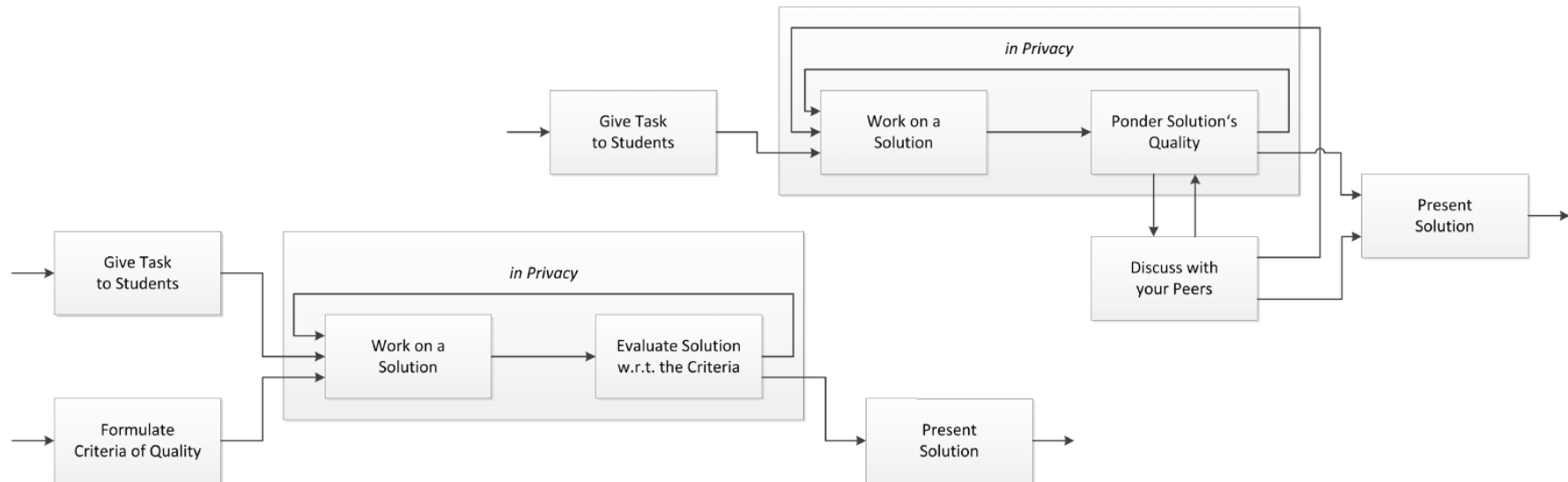
Thus, an appropriate inheritance method can be selected for each key annotation.

## **KE Method # 3: Knowledge Mining for Curriculum Composition**

Data analyses over former students' learning paths in the storyboard by

- 1. Construction** (and successive refinement) **of a decision tree**
  - by using paths that have been gone by students, i.e. paths with a known level of success
- 2. Application of the decision tree**
  - to estimate the success of a planned path

## Storyboard: a simple example



## Storyboards developed so far

- **SIE's study of Tokyo Denki University, Chiba New Town, Japan:**
  - [..\Japan\Japan2006\storyboard\InformationEnvironmentAtTDU\\_2.VSD](..\Japan\Japan2006\storyboard\InformationEnvironmentAtTDU_2.VSD)
- **Course on Intelligent Systems at University of Central Florida, FL, USA**
  - <..\UCFcourse2006\EEL4872 - Intelligent Systems\EEL4872.VSD>
- **Course on Inference Methods at Ilmenau University of Technology, Germany**
  - <..\..\Lehre\AktuelleVorlesungen\StoryboardIM\inferenz.VSD>

## 2.2 Dynamic Storyboard Expansion

Storyboards are nested graphs following a certain type of grammar to be a legal storyboard:

- Episodes are placeholders for **varying** anticipated experiences
- Didactic design is preparation of (various) potential experiences including the specification of preconditions to select a particular one
- If  $G_i$  is a storyboard graph and  $e$  is one of its episodes,  $sub_i(e)$  names all graphs that may be used for the substitution of  $e$
- Using Prolog for reasoning, there is no need to treat graphs as complex data structures. Instead, they are described by predicates
- To avoid ambiguity, it helps to rename nodes that are substituted into another graph such that
  - (1) it remains clear where the nodes are coming from and
  - (2) it is obvious where they are residing now.
- When a certain substitution  $G_i[e \leftarrow G_j]$  takes place, every node  $d \in V_j$  is renamed to  $e.d$ . Accordingly,  $e.V_j$  is shorthand for  $\{e.d \mid d \in V_j\}$ .

To sum up,

**graph expansion is a rewrite relation that terminates with graphs that do not contain any more episodes. Those are formal representations of all the possible human experiences.**

## 2.3 Storyboard Interpretation Technology

= storyboard expansion at execution time while playing and learning

The top level graph of a storyboard has exactly one input node. Executing it means that there are (one or more) pointer(s) indicating the actions to execute:

- (1) In case the action pointed at is a scene, its operational semantics is to be executed such as playing a cut scene or a video, offering buttons or click areas for making a learner's choice, providing documents for download, etc.
- (2) If the pointer indicates an episode, graph expansion is executed. For this purpose, the candidate graphs are found by the function *sub* and the substitution conditions provided by the function *c* are checked. The 1<sup>st</sup> admissible substitution is executed.
- (3) Finally, the pointer is moved forward along admissible edges and, in case there is more than one admissible, split accordingly. In case (2) of substitution, pointers are placed on all substituted input nodes.
- (4) At junctions where edges meet, pointers merge to a common one.

Note that only case (1) is perceived by the human learner/player, whereas (2), (3) and (4) take place in the background.

Adaptivity and personalization is implemented by means of checking the validity of the formulas provided by *c* and *sub*.

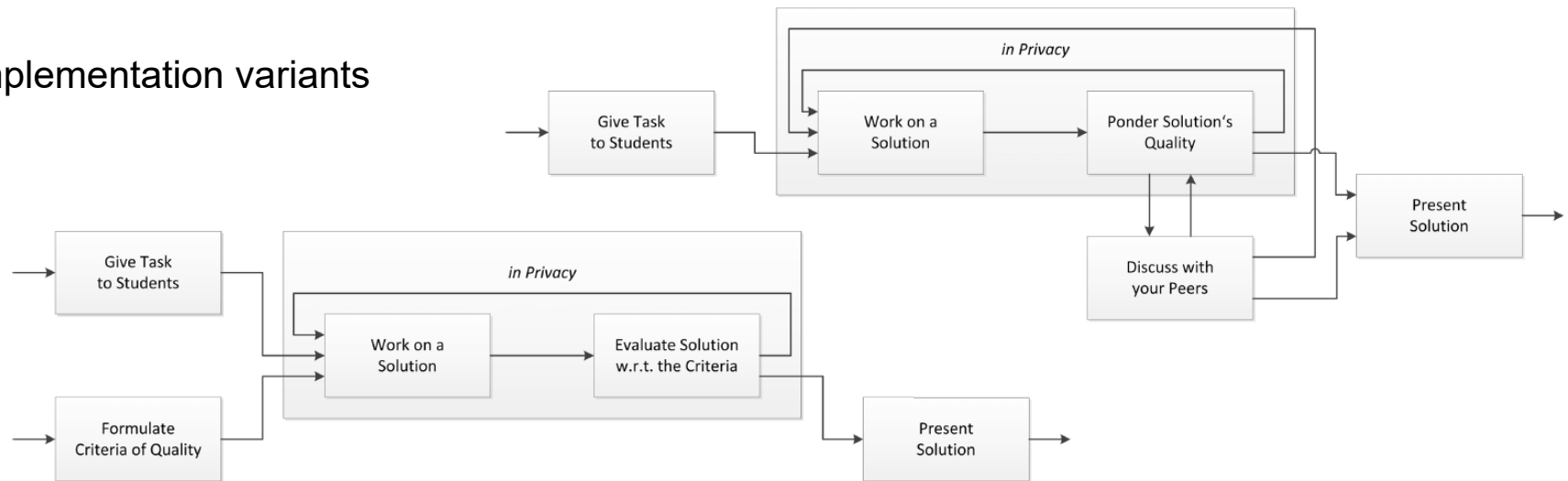
# 3. Plug and Play in Prolog

## 3.1 Static Plug and Play

Storyboard:

- is a hierarchically structured family of graphs  $F = \{G_i\}_{i=1, \dots, k}$  with  $G_1$  being the top level storyboard
- most simple case (we call it **static**): each episode has only one expansion graph within  $\{G_i\}_{i=1, \dots, k}$
- consequently, rewriting  $G_1$  results in a uniquely defined comprehensive storyboard graph that consists of nodes that are scenes, exclusively
- “plug and play” allows trying out different variants of rewriting episodes and comparing these locally encapsulated didactic and/or game play processes by
  - using an alternative episode implementation (**plug**) and
  - checking, whether it makes a difference (**play**)

Example for different implementation variants of the same episode:



## 3.2 Dynamic Plug and Play

Educational gamification may be also seen as a dynamic planning process:

- Storyboards are plans that specify a space of potential future interaction processes learning by playing
- The particular plan to be realized is not known in advance, but unfolds dynamically depending on conditions that may change over time such as
  - context
  - states of the environment possibly including process simulations
  - the emerging learner/player profile
- When a digital storyboard is set up in practice, it remains open whether the one or the other didactic concept meets better both the environmental conditions and the needs and desires of the current users.

## 3.3 Plug and Play with Prolog

### 3.3.1 Plug

Storyboard representation by two predicates:

`node(Graph,Node)`

`edge(Graph,StartNode,EndNode)`

graph expansion requires knowledge about input nodes and output nodes easily provided by to related predicates:

`inputnode(Graph,Node)`

`outputnode(Graph,Node)`

For expanding a graph by substituting another graph into one of its episodes just has to name the first graph, the episode, and the second graph, simply add a fact to the Prolog base and the substitution is done:

`subst(Graph1,Episode,Graph2).`

In the background, the system has knowledge about graph substitution formalized as follows:

`substedge(Graph1,Episode,Graph2,StartNode,EndNode) :-`

`subst(Graph1,Episode,Graph2), edge(Graph1,StartNode,Episode), inputnode(Graph2,EndNode).`

`substedge(Graph1,Episode,Graph2,StartNode,EndNode) :-`

`subst(Graph1,Episode,Graph2), edge(Graph1,Episode,EndNode), outputnode(Graph2,StartNode).`



The system knows furthermore that edges that result from substitutions may be treated as usual edges, because it has the following background knowledge.

```
edge(Graph1,StartNode,EndNode) :-  
    substedge(Graph1,Episode,Graph2,StartNode,EndNode).
```

Because plug and play is a principle that allows for playing with varying alternatives, the ability to unplug one graph substituted before into another for a particular episode is a necessary feature. This is done by retracting all edges that are inserted by substitution of a second graph into a first one for a particular episode:

```
?- retractalldges(Graph1,Episode,Graph2).
```

implemented as follows:

```
retractalldges(Graph1,Episode,Graph2) :-  
    retract(substedge(Graph1,Episode,Graph2,StartNode,EndNode)),!,  
    retractalldges(Graph1,Episode,Graph2).  
retractalldges(Graph1,Episode,Graph2).
```

These are the essentials of Prolog based graph expansion including revision by unplugging.

### 3.3.2 Play

- Due to dynamics, answers to questions like this change over time.
- Simpler, but less reliable is to ask which nodes may be reached by pointers in the course of storyboard interpretation.
- There are some initial facts and rules known to the system, for example to indicate the input nodes where facts as of the top-level graph:  
`pointeron(Graph1InputNode).`  
`pointeron(Node) :-`  
    `edge(NodeBefore,Node), pointeron(NodeBefore), gamma(NodeBefore,Node).`
- There is enormous potentials not yet considered in any storyboarding project by defining meta data:
  - One may easily equip whole graphs and single nodes with meta data such as didactic concepts.
  - On this basis, interesting features of storyboards may be checked automatically such as
    - Occurrence
    - frequency of the occurrences, or
    - alternating occurrence of didactic concepts in potential unfoldings, i.e., normal forms with respect to graph rewritingof a given storyboard.
  - From a higher perspective, one may ask for the occurrence of pattern instances in unfolded graphs.

## 4. Conclusion

To sum up, knowledge about

- the suitability of didactic concepts
- learner and trainee acceptance of playful human-system and peer to peer interaction, and
- the effectiveness of learning and game play amalgamation

emerges throughout educational gamification application and formative evaluation.

***Plug and play*** is the key technology of ***exploratory didactics***.