

Software design document

This document provides an overview of the features that are implemented in the asset 3.3 Communication Scenario Editor.

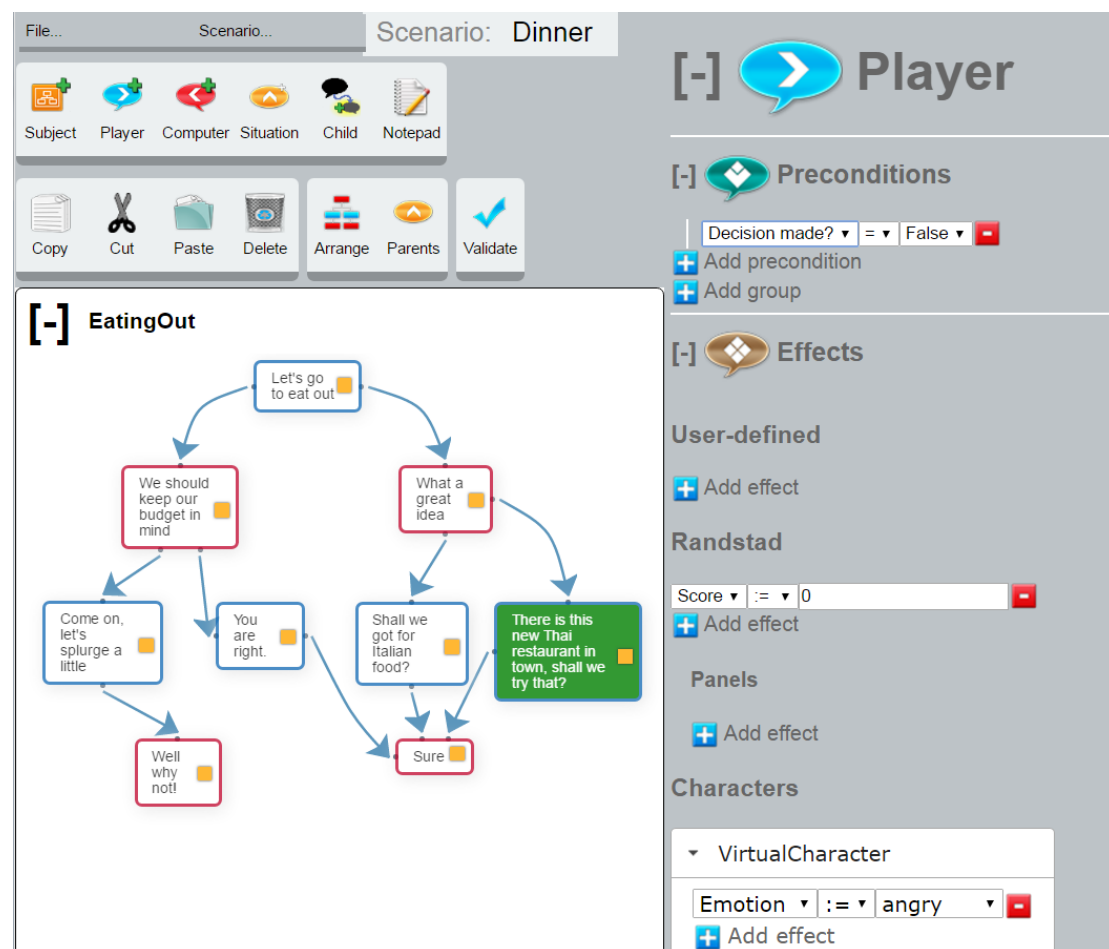
Description

The Communication scenario editor (called scenario editor from now on) is a tool in which a communication expert iteratively develops a communication scenario as a directed acyclic graph of steps. A scenario consists of statements between a player and a virtual character, in which a player choose between various pre-specified options.

The editor balances usability for a non-programming (communications) expert and expressiveness of the constructs in which a scenario can be expressed.

How to use this asset/features

A domain expert develops a scenario in the scenario editor as a graph of steps along with the respective scores and feedback per step. See screen-shot below.



The editor has the following basic features:

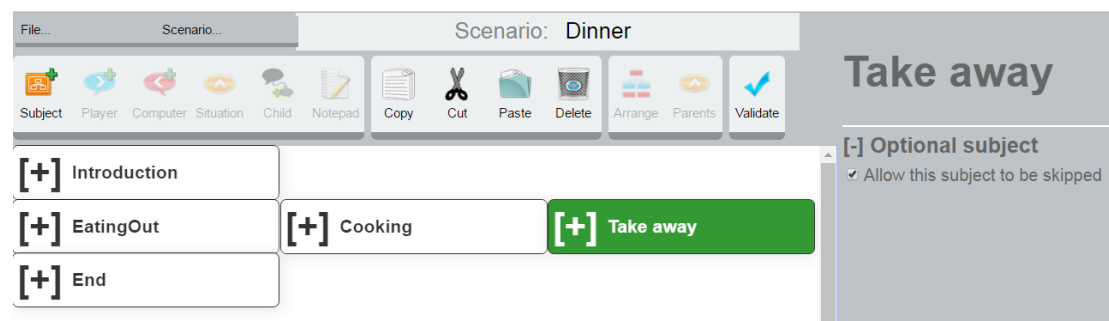
- Develop a conversation between a virtual character (computer node) and a player as a graph. The graph is from top to bottom in time.
- Choices for a player are as multiple player statements from a computer node.

- Each player statement choice leads to a score, and effects like emotions in the virtual character.
- A statement can have a condition.

With the latest release (v4) of the asset, an instance of this editor can be configured catering to the specific needs of a game developer. A game developer can specify virtual character(s), properties (e.g. name of a character) and parameters (e.g. score) specific to their game needs.

- Scores and emotions can be configured as parameters of scenarios, which can be modified at a statement.
- Properties of a scenario can be configured. For example, you can specify the name of a character.

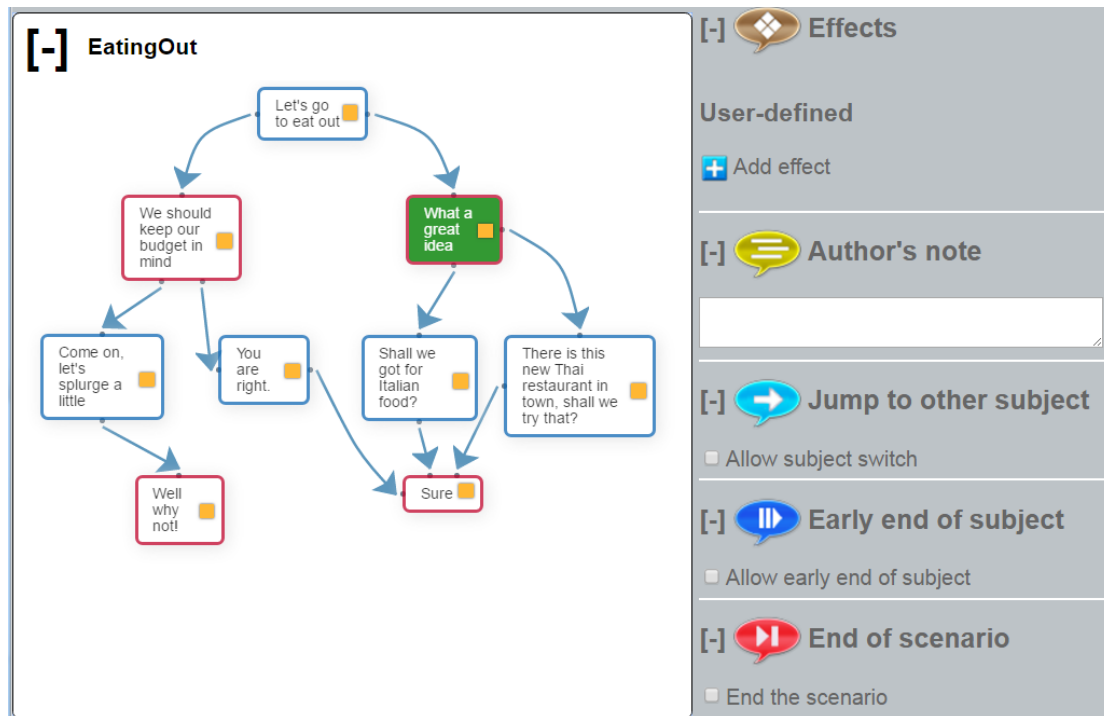
Conversations can be structured at a higher level of abstractions in subjects. See screen-shot below.



A subject on the horizontal level is interleaving with another subject(s) on the same level. In the above figure, 'Eating out', 'Cooking', and 'Take away' are interleaving subjects. During the simulation a player is presented with statement choices from within these interleaving subjects without a predetermined order. This is useful when a player should communicate with a virtual character on multiple subjects, but the order in which statements within these subjects are followed is not important.

A subject may be marked as optional, which means that the subject may be skipped in simulation.

Subject variability is augmented with statement variability. In addition to conditionality in nodes, three features that a scenario author may specify on a computer statement are 'Jump to another subject', 'Early end of scenario' and 'End of scenario'. The interface is depicted in the screen shot below.



A 'jump' indicates whether it is allowed to jump from this node to another node in a subject in this interleave level. An 'early end of subject' indicates that it is allowed to go from this node to a node in one of subjects at the same interleave level and if there are no other subjects in the same interleave, that it is possible to jump to a node in a subject in the next interleave. An 'end of scenario' indicates that the scenario is terminated after this statement.

A combination of these features allows for variability and expressiveness in a scenario.

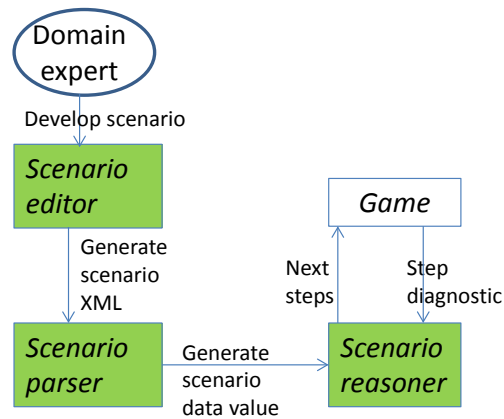
A domain expert can then export valid scenarios in XML format. Scripts may be saved locally and imported to modify and continue development. The output of the editor, a dialogue, is stored as an XML file that follows the schema:

<https://github.com/UUDSL/scenario/blob/master/scenarioLanguage.xsd>

Relation to other asset(s)

The output of the editor can be 'parsed' and 'reasoned' by another asset: 2.2 Step-based competency asset, aka scenario reasoner asset. We have designed the 'Editor' and the 'Assessment' assets to be loosely coupled using a REST architecture. The reasoner asset processes any valid scenario produced according to the schema in the scenario language, using any valid configuration (single Virtual Character /multiple VC's; configurable scores, parameters, properties).

The relation between the assets is shown figuratively below. A game can use the reasoner asset to perform dialogue management.



Technical and quality aspects

The Communication Scenario Editor asset is implemented in Javascript and runs on most browsers, eliminating the need for a native client.

- Demo (<https://www.communicategame.nl/UURAGE/ScenarioEditor/>)
- Repository location: (<https://github.com/UURAGE/ScenarioEditor>), including:
 - Running and building instructions.
 - Deploy instructions.
 - Documentation.